Identifying Trust Properties and Developing Trusted Systems for End-To-End Trust

Jiun Yi Yap

Thesis submitted to Royal Holloway, University of London
for the degree of Doctor of Philosophy

2016

Declaration of Authorship

I, Jiun Yi Yap, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed:

(Jiun Yi Yap)

Date:

Abstract

End-to-end trust is regarded as a game changer for the assurance of distributed and heterogeneous computing environments. It refers to the collection of technologies, user behaviours, implementations and infrastructures that can enable a predictable level of trust in the computing environment. In order to establish end-to-end trust, we first need to identify trust properties that reflects the make up of a trusted system. This is followed by the development of this trusted system that spans hardware, software, people and data.

The first part of the thesis reports our work on identifying trust properties. We first describe a study that looks at how computer users perceive trust notions and the relationship between these perceptions and the stake involved in practical Information Technology scenarios. This work provides an up-to-date understanding of trust notions. Then, to address the challenge of describing the make up of a trusted system, we offer a novel causality-based model. This model represents information about the dependencies between trust notions, capabilities, computing mechanisms and their configurations. We also introduce a new approach to attestation which is founded on the use of provenance data. A complete design of this attestation technique is given. This is followed by building key mechanisms to explore implementation approaches to provenance-based attestation.

The second part of the thesis looks at the challenges of developing trusted systems which contain the building blocks of trust properties. We develop an ontology that describes the capabilities of a computing device secured with the Trusted Platform Module (TPM) 2.0. The aim is to enable experts to share a common understanding of such technologies with developers using a standard vocabulary. We then develop a use scenario of TPM 2.0 and investigate the use of threat modelling on this scenario. Finally, we look at using TPM 2.0, as a building block of trust properties, in a modern system and propose a framework for para-virtualizing TPM 2.0.

Acknowledgment

I would like to thank my supervisor, Dr. Allan Tomlinson, for his support and guidance throughout my PhD program. It was a great challenge for me to go back to school after working in the public sector for 8 years. He helped me with the transition by identifying potential research ideas and giving continuous feedback on my work. He has also introduced me to numerous researchers in Trusted Computing, enabling me to tap on their knowledge. It has been a privilege doing research under his supervision as without him, this thesis would not have been possible. I would also like to thank my advisor, Dr. Stephen D. Wolthusen, who accompanied my development via the annual reviews.

During my stay at Royal Holloway, I was honored to have met many good people. I would like to thank them all and wish them well in their life and careers. Particularly, I would like to thank fellow students from the Information Security Group and Mathematics Department who have brightened up the study environment. In the course of my research work, I was glad to be introduced to David Grawrock of Intel and Dr. Liqun Chen and Dr. Graeme Proudler of HP Labs. I am grateful to them for sharing their knowledge of Trusted Computing. And many thanks to my examiners Assoc. Prof. Andrew Simpson of University of Oxford and Prof. Kenny Paterson of Royal Holloway University of London for their insightful comments on my thesis and making my viva an interesting experience.

Back home, I would like to thank my sponsor for its financial support over the years. From my previous work place, I would like to thank my managers and directors who backed my decision to enroll in this PhD program. I would like to express my deepest gratitude to my colleagues who had to take up my work when I left.

My heartfelt thanks to all my family members for their unconditional love and encouragement. Most importantly, I have to thank my wife for persuading me to take up a PhD program. She has also provided immeasurable care for our boy and me during our stay in the United Kingdom. I simply can't thank her enough.

Contents

Listings **viii**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Key Findings and Contributions . . . 3
1.3 Publications . . . 3
1.4 Thesis Structure . . . 4

I Identifying Trust Properties **8**

2 A Socio-Technical Study on User-Centered Trust Notions and Their Correlation to Stake in Practical Information Technology Scenarios **9**
2.1 Introduction . . . 9
2.2 Literature Review . . . 10
2.3 Research Questions . . . 18
2.4 Scenarios . . . 18
2.5 The Survey . . . 20
2.6 Analyzing the Results . . . 21
2.7 Discussion . . . 27
2.8 Summary . . . 29

3 A Causality-based Model for Describing the Trustworthiness of a Computing Device **30**
3.1 Introduction . . . 30
3.2 A Causality-based Approach . . . 31
3.3 Basic Definitions . . . 32
3.4 Graph Definition . . . 36
3.5 Praxis . . . 38
3.6 Trust Assessment . . . 44
3.7 Related Works . . . 47
3.8 Summary . . . 47

4 Provenance-based Attestation for Trustworthy Computing **49**
4.1 Introduction . . . 49
4.2 Provenance Data as Trust Evidence . . . 50
4.3 Design Goals . . . 51
4.4 A Design for Provenance-based Attestation . . . 52
4.5 Collecting and Representing Provenance Records . . . 53

4.6 Trust Assessment of Provenance Records . . . 58
4.7 Proof of Concept . . . 61
4.8 Threat Modelling of Our Design for Provenance-based Attestation . 63
4.9 A Protocol for Provenance-based Attestation . . . 63
4.10 Related Works . . . 66
4.11 Summary . . . 66

II Developing Trusted Systems 67

5 An Ontology of a Computing Device Secured with Trusted Platform Module 2.0 68
5.1 Introduction . . . 68
5.2 Background . . . 69
5.3 An Ontological Approach . . . 70
5.4 Querying . . . 74
5.5 Related Works . . . 78
5.6 Summary . . . 79

6 Threat Model of a Scenario based on Trusted Platform Module 2.0 Specification 80
6.1 Introduction . . . 80
6.2 Threat Modelling . . . 80
6.3 Description of Scenario . . . 82
6.4 Threat Identification and Mitigation . . . 83
6.5 Related Works . . . 85
6.6 Summary . . . 86

7 Para-Virtualizing the Trusted Platform Module: An Enterprise Framework Based on Version 2.0 Specification 87
7.1 Introduction . . . 87
7.2 More About TPM 2.0 . . . 88
7.3 State of the Art for Para-Virtualizing the TPM . . . 90
7.4 Examining TPM 2.0 Suitability for Para-Virtualizing . . . 92
7.5 Requirements for Para-Virtualizing TPM 2.0 . . . 95
7.6 An Enterprise Framework for Para-Virtualizing TPM 2.0 . . . 95
7.7 Requirements Revisited . . . 99
7.8 Summary . . . 100

8 Conclusion and Future Work 101
8.1 Thesis Summary . . . 101
8.2 Future Work . . . 102
8.3 Conclusion . . . 104

A Appendix A 105

B Appendix B 108

C Appendix C 147

D Appendix D **175**

Bibliography **184**

List of Figures

1.1 Dependencies among chapters of this thesis 7

2.1 Technology acceptance model from [17] 11
2.2 The simplified user centered trust model from [32] 12
2.3 Relationship between trust, stake and risk drawn after [82] 16
2.4 Relationship between trust and trustworthiness drawn after [82] 17
2.5 Willingness to share different types of personal information in scenario one. 22
2.6 Willingness to share different types of personal information in scenario two. 25

3.1 Definition of causality-based model. 35
3.2 Defining the causal graph. 37
3.3 Pictorial representation of the vertices and edges in the causal graph model. 38
3.4 Pictorial representation of a sample causal graph. 44
3.5 Example of a trust policy. 46

4.1 A design of provenance-based attestation. 52
4.2 A graph describing the PROV data model redrawn from [54]. 54
4.3 Flowchart for upgrading iptables using rpm. 55
4.4 Provenance graph of upgrading iptables. 56
4.5 A screen shot of System Tap running a monitoring script. 61

5.1 Graphical representation of the ontology based description. 72
5.2 SPARQL query for the *Confidentiality* class. 75
5.3 SPARQL query for the *AccessControl* class. 77
5.4 SPARQL query for the *Attestation* class. 77

6.1 STRIDE-per-element matrix from [34]. 81
6.2 To encrypt symmetric key for group share. 82
6.3 To recover symmetric key for group share. 83
6.4 DFD for encrypting symmetric key (left) and for recovering symmetric key (right). 86

7.1 Architecture for para-virtualizing TPM sharing from [21]. 91
7.2 Layout of the multi-context TPM from [85]. 91
7.3 TPM control structure from [85]. 92
7.4 TPM 2.0 key distribution for multiple VM 93
7.5 Enterprise framework for para-virtualizing TPM 2.0. 96

List of Tables

2.1 Subjective trust notions. 13
2.2 Objective trust notions. 15
2.3 Demographics of the two samples. 22
2.4 Ranking of subjective notions in scenario one. 23
2.5 Ranking of objective notions in scenario one. 24
2.6 Ranking of subjective notions in scenario two. 26
2.7 Ranking of objective notions in scenario two. 26

4.1 Threats and mitigations to provenance-based attestation. 64

5.1 Result for SPARQL query on the *Confidentiality* class. 75
5.2 Result for SPARQL query on the *AccessControl* class. 77

6.1 Threats and mitigations to use scenario. 85

Listings

3.1 XML schema of causal graph data model. 39
3.2 XML representation of the sample causal graph. 41
3.3 Specification of the basic assessment rule. 44
3.4 XQuery command to check for a specific mechanisms that ″DiskLocker″ calls on. 45
4.1 XML based provenance record for patching iptables. 57
4.2 Rule specification grammar for trustworthy evaluation of provenance record. 59
4.3 A set of dependency and attribute rules. 60
4.4 Dependency rule expressed as XQuery. 62
4.5 Attribute rule expressed as XQuery. 62
4.6 Using TPM to generate digital signature of a provenance record. . . . 65
4.7 Using TPM to verify digital signature of a provenance record. 65
5.1 Description of the *Confidentiality* class in RDF/XML format. 75
5.2 Description of the *AccessControl* class in RDF/XML format. 76
5.3 Description of the *Attestation* class in RDF/XML format. 77

List of Abbreviations

CCE	Common Configuration Enumeration
CPE	Common Platform Enumeration
DFD	Data Flow Diagram
DNF	Disjunctive Normal Form
I/O	Input and Output
IT	Information Technology
IV	Initialization Vector
MAP	Metadata Access Point
NIST	National Institute of Standards and Technology
NV	Non Volatile
OPM	Open Provenance Model
OS	Operating System
OWL	Web Ontology Language
PCR	Plaform Configuration Register
RDF	Resource Description Framework
RNG	Random Number Generator
TAM	Technology Acceptance Model
TCB	Trusted Computing Base
TCG	Trusted Computing Group
TLS	Transport Layer Security
TNC	Trusted Network Connect
TPM	Trusted Platform Module
Turtle	Terse RDF Triple Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VM	Virtual Machine
VMM	Virtual Machine Monitor
W3C	World Wide Web Consortium
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

Chapter 1

Introduction

1.1 Motivation

Modern computing devices are diverse and interconnected by dynamic and heterogeneous networks. In the 2009 National Cyber Leap Year Summit, the participating researchers reported that end-to-end trust will be a game changing technology when deployed in this type of computing environment [14]. They defined end-to-end trust as a collection of technologies, behaviours, implementations and infrastructure that can enable a predictable level of trustworthiness in the computing environment.

The identification of trust properties can be used to determine the trustworthiness of a system. This ability is considered as a key to the establishment of end-to-end trust. The benefit of this is that any other computing device can select its mode of participation in a computer network according to the level of trustworthiness offered by the corresponding computing devices. In the same vein, Grawrock et al. explained that there is a need for computing devices to communicate their trustworthiness so that the involved parties can understand and manage security risks [94]. Grawrock articulated on this issue again at the 2013 European Trusted Infrastructure and Systems School [1]. He lectured specifically on the need for a trust language that could describe the trustworthiness of a computing device. He noted that the level of trust is not the same for different tasks. On the other hand, the make up of a computing device determines its level of trustworthiness and consequently the tasks it is trusted to perform. The trust language should have grammar to support the descriptive property, and rules are necessary as free form makes parsing difficult.

Meanwhile, property-based attestation was proposed by Sadeghi and Stuble [74]. The main concept is that attestation should verify if a computer possesses properties to fulfill certain requirements of the party who asks for attestation. The authors define that the trust property of a computing device describes an aspect of the behaviour of that computing device regarding certain requirements, such as confidentiality. However, this approach is challenged by the fuzzy definition of property. In addition, the authors did not discuss how this property can be evaluated to ascertain the trustworthiness of the computing device.

In addition, a literature review was carried out by Nagarajan et al. [55] on property based attestation. In that paper, the authors analyzed the various works and summarized that the main challenge was to define what is meant to be a property. The authors also wrote that a property can be described at different level of granularity. For example, data confidentiality in a card payment machine is pro-

[1] http://www.iaik.tugraz.at/content/about iaik/events/ETISS INTRUST 2013/

vided by an encryption function which employs a certain cryptographic algorithm. Moreover, the definition of a property is closely bound to the application that the property is defined for. The property can change as the state of the computer system changes. There is also the consideration that there are dependencies among properties. Hence, during attestation, the list of dependent properties has to be evaluated too. As the paper is a literature review, the authors did not go into describing how these properties make up a trusted system. Nevertheless, the paper provides a very useful analysis on trust properties.

Thus, some questions on identifying trust properties arise: What is the understanding of trust by computer users? How can we describe the trustworthiness of a computing device? Can this description be evaluated? What evidence can be provided to convey assurance in the trustworthiness of a computing device? Can we evaluate this evidence? In the first part of this thesis, we study and address these questions.

The second part of this thesis looks at the development of trusted systems which contain the building blocks of trust properties. The participating researchers of the 2009 National Cyber Leap Year Summit stated that hardware can be the root of trust in a computing environment. Thus, a system that has components which are developed and configured to leverage trusted hardware can be considered as a trusted system [66]. Moreover, such trusted hardware is also regarded as a building block for trust properties.

On the research front, considerable effort was directed towards building a trusted system based on the Trusted Platform Module (TPM). The TPM is a device that is specified by the Trusted Computing Group (TCG). The TPM is designed to improve the trust in computing devices by offering certain functionalities such as cryptographic engine, secure storage, digital certification and trusted reporting of the identity and state of its host computing device. TPM 2.0 is the latest specification from TCG [92].

One such project which aimed to build trusted systems was carried out by Kuhlmann et al. [42]. In this project, prototypes of trusted systems were developed. For example, in the demonstration for home banking, a specialized virtual machine was launched to host the home banking client. This virtual machine will be attested by the bank to prevent impersonation. Although the TPM provides unique capabilities which can benefit computer users, a panel of industry experts at the Trusted Computing Conference 2013 pointed out that its uptake was mainly limited to high end computing devices such as those designed for the commercial environment [2]. One reason given was that there was a lack of consumer use scenarios.

Consequently, some problems on the development of trusted systems emerge: How to share information on trusted hardware so that developers can use it? What method can be used to identify threats to use scenarios of trusted hardware? What applications can benefit from using trusted hardware? In the second part of this thesis, we investigate and propose solutions to these problems.

1.2 Key Findings and Contributions

This thesis uses several approaches to develop solutions for the questions raised in Section 1.1. In so doing, the thesis makes the following findings and contributions.

- We uncovered how computer users perceive subjective and objective trust notions in various scenarios. We also detected correlations between the stake involved in a scenario and the importance of these trust notions. Consequently, we contributed an updated understanding of trust.
- To describe the properties that make up a trusted system, we showed how this trust language can be based on the concept of causality. The description includes information about the dependencies between trust notions, capabilities, computing mechanisms and their configurations. To support practical application, we transformed this causality-based description to a graph and demonstrated its implementation. This work brings together two separate domains of research in trust: abstract trust notions and technical trust primitives. As a result, we contributed a causality-based model for describing the properties of a trusted system and the specification of rules used for its evaluation
- We argued for the use of provenance data as trust evidence. To advance this idea, we developed a design that uses provenance data of software components in the attestation of a trustworthy computing device. This design includes the specification of rules used for evaluating provenance data. Therefore, we contributed an attestation technique based on provenance data of computer software components.
- We found that the ontological approach is suitable for producing a standard vocabulary for experts to share with developers the common understanding of a particular class of trusted system. Subsequently, we contributed an ontology of a computing device secured with TPM 2.0
- To use TPM 2.0 as a building block of a trusted system, we found that threat modelling can be used as a tool to analyse use scenarios of TPM 2.0. Meanwhile, we also found that TPM 2.0 can be para-virtualised. On this point, we contributed a para-virtualisation framework for TPM 2.0.

1.3 Publications

The contributions in this thesis are described in the following publications.

- Jiun Yi Yap and Allan Tomlinson. Socio-Technical Study on User-Centered Trust Notions and Their Correlation to Stake in Practical Information Technology Scenarios. In Proceedings of the 6th ASE International Conference on Privacy, Security and Trust, 14-16 December 2014.
- Jiun Yi Yap and Allan Tomlinson. A Causality-based Model for Describing the Trustworthiness of a Computing Device. In Proceedings of the 7th International Conference on Trusted Systems, 7-8 December 2015.

- Jiun Yi Yap and Allan Tomlinson. Provenance-based Attestation for Trustworthy Computing. In Proceedings of the 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 20-22 August 2015.
- Jiun Yi Yap and Allan Tomlinson. An Ontology of a Computing Device Secured with Trusted Platform Module 2.0. RHUL-ISG-2016-2 (Information Security Group, Royal Holloway, University of London, 2016).
- Jiun Yi Yap and Allan Tomlinson. Threat Model of a Scenario Based on Trusted Platform Module 2.0 Specification. In Workshop on Web Applications and Secure Hardware, 20 June 2013.
- Jiun Yi Yap and Allan Tomlinson. Para-Virtualizing the Trusted Platform Module: An Enterprise Framework Based on Version 2.0 Specification. In Proceedings of the 5th International Conference on Trusted Systems, 4-5 December 2013.

1.4 Thesis Structure

This thesis is organised into two parts. Part I contains Chapters 2, 3 and 4, which focus on the identification of trust properties. In Chapter 2, to establish the understanding of trust notions in this thesis, we describe a socio-technical study on user-centered trust notions. Addressing the challenge of describing the trustworthiness of a computing device, we present, in Chapter 3, a novel causality-based model to represent information about the dependencies between trust notions, capabilities, computing mechanism and configurations. This is followed by Chapter 4 where we address the question of the assurance in the trustworthiness of a computing device by introducing a new approach to attestation using the provenance data of the computer components.

Part II consists of Chapters 5, 6 and 7, and they deal with the development of a trusted system which uses the TPM 2.0 as a building block. In Chapter 5, we present an ontology that describes the capabilities of a computing device secured with TPM 2.0. The aim is to have a standard vocabulary for experts to share information on such technologies with developers of trusted systems. This is followed by Chapter 6 where we develop a use scenario based on TPM 2.0 and study about the use of threat modelling. In Chapter 7, we offer a framework for para-virtualizing as a potential application of TPM 2.0.

As this thesis spans multiple research projects, the background material relevant for their understanding and the literature reviews are woven into their respective individual chapters. The following paragraphs provide more details about these chapters.

Chapter 2. To establish the understanding of trust notions in this thesis, we conducted a socio-technical study on user-centered trust notions and their correlations to stake in practical information technology scenarios. From literature reviews, the study captured a list of subjective and objective trust notions and looked at a conceptual relationship between trust, stake and risk. We then proposed an approach

to investigate the importance of these trust notions and their correlation with stake. For the purpose of the study, an online survey was commissioned and it provided the respondents with scenarios based on a mass market laptop computer and Internet of Things devices. It asked questions on the type of information for sharing online to establish the stake and the importance ranking of the subjective and objective trust notions. The results were analyzed and hypothesis tests were carried out using statistical methods. We uncovered how the respondents rank the importance of various trust notions in different scenarios and detected correlations between stake and the perceived importance of the subjective and objective trust notions. This work was partly funded by the University of London Postgraduate Research Study Cost Scheme. This chapter is based on our publication "Socio-Technical Study on User-Centered Trust Notions and Their Correlation to Stake in Practical Information Technology Scenarios".

Chapter 3. With the understanding of trust notions from Chapter 2, we present in this chapter a novel causality-based model for describing the trustworthiness of a computing device. The description includes information about the dependencies between trust notions, capabilities, computing mechanisms and their configurations. We will show in this chapter what could be the grammar of Grawrock's trust language described in Section 1.1 and we explain the structure of this trust language. In this work, the concept of causality within the model was defined first. This involved detailing the semantic meaning of the terms used in the model. A pictorial representation was then developed to show the causal dependencies as a graph. This step specified the vertices and edges used in the causal graph. To implement the causality-based model, the causal graph was translated into an eXtensible Markup Language schema and added to the Metadata Access Point database server of the Trusted Network Connect open architecture. Finally, the trust assessment of the causal graph was explained. The work presented in this chapter appears in our publication "A Causality-based Model for Describing the Trustworthiness of a Computing Device".

Chapter 4. Having developed a description for trustworthiness of a computing device, the next step is to attest to the components identified in the description. We present a new approach to attestation that is founded on provenance data of its key components. The prevailing method of attestation relies on comparing integrity measurements of the key components of a computer against a reference database of trustworthy integrity measurements. An integrity measurement is obtained by passing the binary of a component through a hash function but this value carries little information unless there is a reference database. On the other hand, the semantics of provenance contain more details: There is expressive information such as the component's history and its causal dependencies with other elements of a computer. Hence, we argue that provenance data can be used as evidence of trustworthiness during attestation. We begin by describing a complete design for provenance-based attestation. The design development was guided by goals and it covers all the phases of this approach. We discuss collecting provenance data and using the PROV data model [54] to represent provenance data. This is a new application of the PROV data model which is typically used in the semantic web. To evaluate the provenance data for trustworthiness of the component it represents, we developed a rule specification grammar and provided a discourse on using the

rules. We then built the key mechanisms of this form of attestation by exploring approaches to capture provenance data and looking at transforming the trust evaluation rules to XQuery language before running the rules against an XML based record of provenance data. Finally, the design was analyzed using threat modelling. This chapter is based on our publication "Provenance-based Attestation for Trustworthy Computing".

Chapter 5. The second part of this thesis looks at the development of trusted systems which contain the building blocks of trust properties. We first work on using an ontology as a standard vocabulary for experts to share a common understanding of trusted systems with developers. To this end, we contribute an ontology for describing a class of computing device secured with Trusted Platform Module (TPM) version 2.0. This ontology represents information about the capability of the computing device, TPM 2.0, and notions of trust we have obtained from Chapter 2. They are characterized at different abstract levels. The ontology is based on the Web Ontology Language (OWL) and recorded in Resource Description Framework (RDF) / eXtensible Markup Language (XML) format. A developer can review the ontology-based description by asking competency questions. Finally, we demonstrate how to translate the competency questions into SPARQL queries. The work presented in this chapter appears in our technical report "An Ontology of a Computing Device Secured with Trusted Platform Module 2.0".

Chapter 6. In this chapter, we produce a use scenario that describes the use of TPM 2.0 cryptographic functions to share data. This is followed by the use of Microsoft's security development lifecycle threat modelling tool to construct a threat model of this use scenario. The threats to each element in the model are analysed and the appropriate mitigations are worked out. Our publication "Threat Model of a Scenario Based on Trusted Platform Module 2.0 Specification" forms the basis of this chapter.

Chapter 7. We now consider how TPM 2.0, as a building block for trust properties, can be used in a modern system. Virtualization is a fundamental technology that is widely used in Enterprise IT infrastructures. Users of virtualization technology need some level of assurance about the expected behavior of a virtual machine (VM) and its ability to protect confidential information from unauthorized disclosure. The TPM offers security properties that can be leveraged by the users of virtualization technology to increase the protection of the system and data from cyber security threats [77]. In this chapter, we contribute a framework for para-virtualizing the TPM 2.0. The framework covers the design of a para-virtualized TPM 2.0 and the considerations when deploying it for use in an Enterprise Information Technology infrastructure. To develop this framework, a quick study of the TPM 2.0 specification was undertaken and a survey of para-virtualizing TPM techniques was carried out. The study found that TPM 2.0 core functions are suitable for para-virtualization. A set of requirements was then developed to guide the design of this framework. The framework includes components to support the para-virtualized TPM. The framework also covers external components that are essential for the proper functioning of the para-virtualized TPM in an Enterprise IT environment. The work presented in this chapter appears in our publication "Para-Virtualizing the Trusted Platform Module: An Enterprise Framework Based on Version 2.0 Specification". It was given the best paper award by the conference

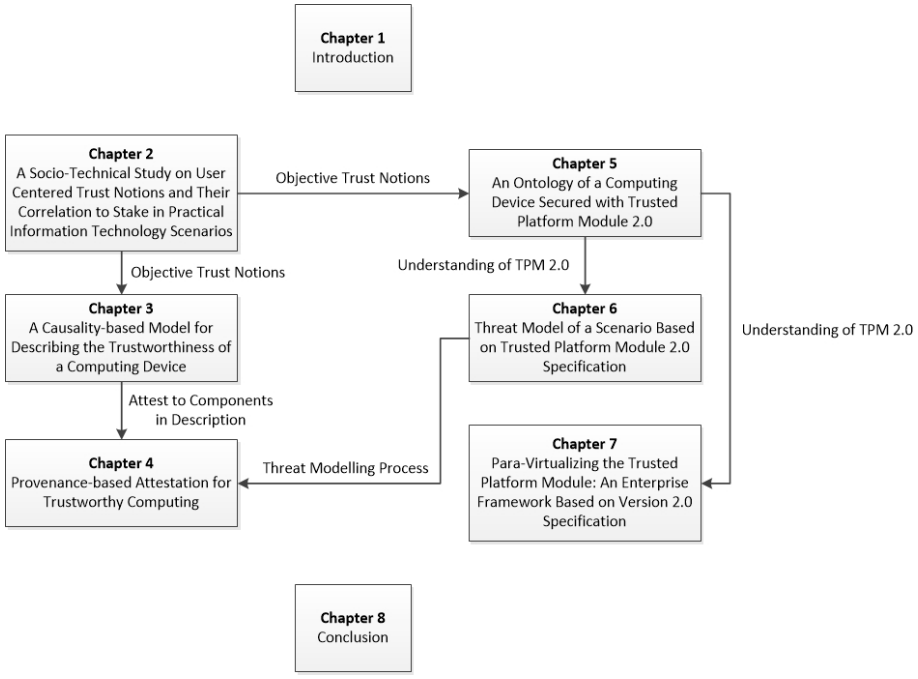


Figure 1.1: Dependencies among chapters of this thesis

organisers.

Chapter 8. We conclude this thesis by providing final remarks and discussing the plethora of research still to be undertaken in the course of these works.

Although Chapters 2 to 7 present individual works, they are not entirely independent and do exhibit associations. For example, the threat modelling process in Chapter 6 is used to analyse the provenance-based attestation in Chapter 4. The objective trust notions from Chapter 2 are used to guide the development of the causality-based model in Chapter 3 and the ontology in Chapter 5. The ontology in Chapter 5 is in turn referenced when developing the materials for Chapter 6 and 7. Lastly, Chapter 4 describes an attestation technique that can be used to vouch for the trust description of Chapter 5. These relations are depicted in Figure 1.1.

Part I

Identifying Trust Properties

Chapter 2

A Socio-Technical Study on User-Centered Trust Notions and Their Correlation to Stake in Practical Information Technology Scenarios

2.1 Introduction

In the context of human and computer interaction, it is often the case that the human computer user has to trust a computing device to carry out a specific task involving certain stake. A key factor that affects this behavior is the properties of the computing device that satisfy the human computer user's requirement for trustworthiness. This in turn relates to how a human computer user understands trust. In this thesis, we use the term "trust notion" to refer to the properties of a computing device that a human computer user considers when he is deciding whether to trust it to carry out a given task. Another key factor is the stake involved in this task. Stake will refer to the value of information revealed when a task is carried out by a computing device. The computer user can decide what stake is acceptable with respect to the properties of the computing device and the scenario.

The user-centered point of view of the trust in a computing device is more subtle than "a computer user should not use that computing device if he cannot trust it" [39]. Trust is not a purely technical property but rather a socio-technical quality that includes computer users' behavior with regards to technology and the underlining cognitive and psychological factors [15]. Moreover, the stake involve in the task can influence a computer user's attitude to the properties offered by the computing device [82]. This is an important consideration in Trust Management. For example, if the task involves information that is considered as high value, then a computer user is likely to require the computing device to possess a configuration that gives a high level of protection.

We would like to obtain an up-to-date understanding of how computer users view the social and technical aspects of trust in practical scenarios. We would also like to gain an insight into the connection between stake and computer users' perceptions of trust notions. This knowledge will provide direction when investigating trust concepts and the use of trustworthy computing technologies. Thus, a socio-technical study was carried out to examine how computer users perceive these trust notions and the association of trust notions to the stake involved in practical scenarios. The social aspect of the study examined how these mental and experiential attributes influence a computer user's behavior in trusting a computing device. The technical aspect of the study looked at how a computer user perceives certain technical properties such as those related to information security, trusted computing and security standards.

Section 2.2 reviews related work. It captures the trust notions that are covered in this socio-technical study and examines the conceptual relationship between trust, stake and risk. This section also describes the approach taken to study these concepts in practice. Section 2.3 sets out the research questions and Section 2.4 describes the scenarios that portray the stakes involved. Section 2.5 describes the development of an online survey to collect data for this study. Section 2.6 examines the data collected from the online survey using statistical analysis and hypothesis testing. Section 2.7 discusses how the survey data helps to answer the research questions. The summary of this chapter is in Section 2.8. In addition, secondary analyses of how different demographic groups perceive the subjective and objective trust notions in the two scenarios are given in Appendix A.

2.2 Literature Review

The concept of trust is so diverse in different domains and contexts that it is not meaningful to articulate it without a focus. In this socio-technical study, trust is examined in the context of human and computer interaction which is typical in the domain of Information Technology. A number of user-centered trust models have been proposed. Corritor et al. propose a model for online trust [16]. Their trust model is based on research from human-computer interaction and it defines two categories of factors that give rise to trust. In the category of perceived factors, the authors describe how credibility, ease of use and risk can impact a computer user's degree of trust in a website. On the other hand, in the category of external factors, the authors describe the influence on trust by physical and psychological causes. Meanwhile, Wang and Emurian suggest a framework for online trust [95]. The framework presents the four considerations of graphic design, structure design, content design and social-cue design. The authors explain that these considerations are aimed to gain a high level of trust from the computer users. However, [16] and [95] focus on the social aspect of trust and they do not address the technical aspect of trust. Thus, to examine both the social and technical aspect of trust, we turn to the user-centered trust model proposed by Hasan et al. [32] which suggests that trusting beliefs form attitudes that lead to trusting behavior. Thereafter, we look at the conceptual relationship of trust, stake and risk by Solhauf et al. [82]. Their literature reviews are summarised in Section 2.2.1 and 2.2.2. As defined earlier, the social aspect of the study examines how mental and experiential attributes influence a computer user's behavior of trusting a computing device and we use the term "subjective trust notion" to refer to these attributes. On the other hand, the technical aspects of the study looked at how a computer user perceives certain technical properties such as those related to information security, trusted computing and security standards, and the term "objective trust notion" is used in this chapter to refer to those properties.

2.2.1 User Centered Trust Model

The framing of this study is based on the conceptual user centered trust model proposed by Hasan et al. [32]. In their paper, the authors described how the Technology Acceptance Model (TAM) [17] can be integrated with various trust concepts

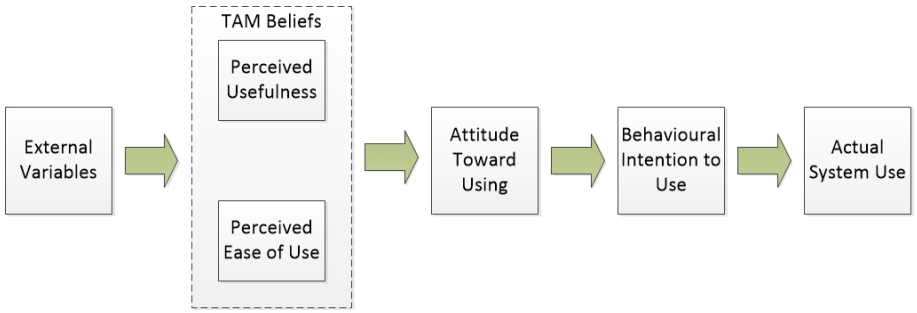


Figure 2.1: Technology acceptance model from [17]

they gathered from literature reviews. The main idea of the TAM is that system use is a response that can be explained or predicted by user motivation. This in turn is directly affected by external variables, which could be the user's education level, gender, mindset towards adopting new technology and stake and risk involved in a transaction. We will look at the external variables of stake and risk in Section 2.2.2. User motivation consists of the following factors: beliefs, attitude toward using and behavioral intention. The TAM explains that the components of beliefs include the perceived usefulness and perceived ease of use. Perceived ease of use is the degree to which an individual believes that using a particular system would be free of physical and mental effort while perceived usefulness refers to the degree to which an individual believes that using a particular system would enhance his work performance. The attitude of the user is a major determinant of whether the user will actually use or reject the system and behavioral intention is a measure of one's intention to perform a behavior. Figure 2.1 shows the TAM.

The TAM can be explained by the Theory of Reasoned Action [23], which suggests that behavior is driven by intentions and intentions are a function of an individual's attitude. In turn, these attitudes are derived from beliefs. With this understanding in mind, Hasan et al. then carried out literature reviews and highlighted the work of Benamati et al. [7] and McKnight et al. [51]. The authors deduce from the literature reviews that beliefs can be categorized into "trusting beliefs in technology" and "trusting beliefs in vendors". The trusting beliefs in technology refers to the technical properties that a computer user thinks will enable a computing device to accomplish the given task. The trusting beliefs in vendors refers to the conduct that people in the organization, that produce the computing device, display in order to impress the computer user, who is a consumer of the computing device. Based on their analysis, the authors proposed a hypothetical user centered trust model and the simplified illustration of this model is shown in Figure 2.2. Essentially, this user-centered trust model proposes that beliefs, which consist of the TAM beliefs, trusting beliefs in technology and trusting beliefs in vendor, and in accordance to the Theory of Reasoned Action, give rise to attitudes that lead the computer user to form the intention to trust a computing device and eventually result in a behavior that manifests as using the computing device to carry out a given task.

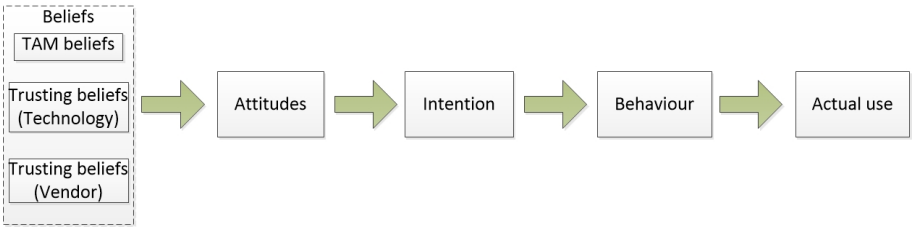


Figure 2.2: The simplified user centered trust model from [32]

2.2.2 Subjective Trust Notions

We argue that notions associated with TAM beliefs and trusting beliefs in vendors can differ from one person to another and are subjective. For example, each computer user has his belief about how a computing device can enhance his job performance. Another example is that one individual's experience with the vendor can be different from another's. On the other hand, notions related to trusting beliefs in technology are based on technical attributes and hence objective. Therefore, we can broadly classify these trust notions from the user-centered trust model as either subjective or objective. The subjective trust notions to be examined in this study are proposed to consist of those shown in Table 2.1. These notions are derived from the notions described by Hasan et al. for TAM beliefs and trusting beliefs in vendors.

2.2.3 Objective Trust Notions

For the purpose of a more encompassing socio-technical study on user centered trust notions, we propose that the trusting beliefs in technology cover notions related to the attributes of information security and trusted computing. This is because trust, as discussed in the foregoing, and security are not orthogonal. Trusted components are used to build secure systems. On the other hand, security properties are often evaluated as part of the process of establishing trust in a computing system. This association of trust and security is articulated in the Trusted Computer System Evaluation Criteria (Orange Book) of the United States of America Department of Defense in 1985 [71]. Therefore, our study includes notions related to attributes of information security such as confidentiality, integrity and availability. These notions are described in the ISO 27001 standard [37]. In addition, the study looks at what computer users think of the importance of security standards that prescribe guidelines for the secure development and configuration of computing devices.

One central piece of Information Security technology is the Trusted Platform Module (TPM) specified by the Trusted Computing Group [92]. The TPM is a device which is typically bound to another computing device to provide trusted computing functionality. One aspect of this functionality is the integrity measurement of components involved in the booting of the computing device. When the integrity measurements match predefined trustworthy values, it can be said that the computing device will behave in a trustworthy manner. This functionality conveys the notion of expected behavior and authenticity.

Furthermore, the TPM has a unique key that can be used to identify itself to

Table 2.1: Subjective trust notions.

S/N	Notion	Description
1	The computing device is easy to use.	The degree to which a computer user believes that using the computing device will be free from physical and mental effort. This notion is related to TAM.
2	The computing device is useful for this scenario.	The degree to which a computer user believes that the computing device will enhance his job performance. This notion is related to TAM.
3	The aesthetics of the computing device suit this scenario.	The emotional attitude that a beautiful or attractive casing or interface gives a positive impression. Hassan et al. cited the work by Lindgaard et al. [46] who suggested that visual appeal dominates first impression judgment of trustworthiness.
4	Vendor's technical competence at solving your problem.	The degree to which a computer user perceives that the vendor is in possession of the necessary knowledge and skills to resolve a technical problem. This notion is part of the trusting belief in technology and the authors attributed this to the work by Mayer et al. [50] and Li et al. [45].
5	Vendor's interest in your wellbeing when you encounter a difficulty.	The belief that the vendor is interested in the wellbeing of the user without ulterior motive. This notion is part of the trusting belief in technology and the authors attributed this to the work by Mayer et al. [50] and Li et al. [45].
6	Vendor's ability to tell you the truth and act ethically when a new issue is discovered.	The belief that the vendor tells the truth and acts ethically. This notion is part of the trusting belief in technology and the authors attributed this to the work by Mayer et al. [50] and Li et al. [45].

a requestor. Identity is an important factor affecting trust. During remote attestation, the TPM can provide the requestor this identification together with the integrity measurements as a proof of its identity and state of trustworthiness. In addition, the latest specification of TPM has a cryptographic engine that is able to carry out cryptographic operations using widely used symmetric and asymmetric algorithms.

Although the TPM provides unique capabilities which can benefit computer users, its uptake is mainly limited to high end computing devices such as those designed for the commercial environment. A panel of industry experts discussed this at the Trusted Computing Conference 2013 [2] and pointed out that there is a lack of consumer use cases which may hamper the uptake of the TPM. Therefore, the outcome of our study into TPM related notions such as expected behavior, identity and authenticity can also aid the development of TPM based consumer use cases.

A new trustworthy computing research area is the use of provenance-based models for reasoning about a system's ability to satisfy trust properties of interest [56]. The main concept behind the provenance-based model is the provision of origin and change history of hardware or software components to assist with trust evaluation. It is interesting to examine what computer users think of this property of provenance as an objective trust notion.

Consequently, the objective trust notions to be examined in this study are proposed to consist of those described in Table 2.2 which relate to properties of the device.

We now have captured a list of subjective and objective trust notions for examination in this study. As explained by the user-centered trust model of Hasan et al., these notions are the beliefs that give rise to attitudes which lead the computer user to form the intention to trust a computing device and subsequently the act of using it to carry out a task. In our socio-technical study, we will focus on investigating the degree of importance of each of these trust notions to the computer users when they have to trust a computing device to carry out a task in a particular scenario.

2.2.4 Relationship between trust, stake and risk

From Section 2.2.1, we understand that stake and risk are considered as external variables in the TAM. In this sub-section, we will review the conceptual model from Solhauf et al. [82] that relates the various facets of trust, stake and risk. We will then propose an approach to investigate this concept by linking the definition of trust and trustworthiness in this model to the subjective and objective trust notions explained in the previous sub-section on the user-centered trust model.

Solhauf et al. explain that trust is a relationship between a trustor and a trustee where the former places trust in the latter. They define "trust" as the subjective probability by which the trustor expects the trustee to perform a given action on which their welfare depends. The important property of "trust" in this model is the aspect of belief, i.e. the trustor's subjective probability estimation. This level of "trust" is a probability that ranges from 0 (complete distrust) to 1 (complete trust). To extend this definition to the user centered trust model of Hassan et al., we propose that this understanding of "trust" can be extrapolated to include subjective trust.

Table 2.2: Objective trust notions.

S/N	Notion	Description
1	Always work in the same way as before (expected behaviour)	This is supported by TPM's boot up integrity measurement .
2	Provides identity and authenticity of the computing device.	This is supported by TPM's unique key and integrity measurement.
3	Ensure confidentiality of data.	This is an attribute of information security.
4	Ensure integrity of data.	This is an attribute of information security.
5	Provides the origin and change history of its hardware and software components.	This is a new research topic that aims to use provenance data of trustworthy components.
6	Required function or data is always available.	This is an attribute of information security and is related to the notion of functionality and reliability described under the trusting beliefs in technology of the user centered trust model.
7	Is certified to meet certain security standard, for example, Common Criteria.	Security standards prescribe guides for implementing information security.
8	Provides identity and authenticity of the corresponding party.	This is supported by TPM's unique key and integrity measurement.

Solhauf et al. argue that the level of risk is balanced against the stake involved and the trust in the trustee. Stake refers to the value involved in the transaction, at a certain level of risk, between the trustor and trustee. This relationship between trust, stake and risk is illustrated in Figure 2.3.

As an example of using Figure 2.3, a trustor is assumed to only accept low risk in a specific context. If the value in the transaction is s2, the trustor must trust the trustee with a value of "trust" $\geq$ t2. Symmetrically, if the "trust" value is determined to be t2, the trustor will proceed with transactions in which the stake is $\leq$ s2. If the trustor does not accept high risk, then all transactions in which "trust" value is t1 and the stake value is $\geq$ s1 are unacceptable. In the same way, if the stake is s1, all transactions in which the "trust" value is $\leq$ t1 are unacceptable.

Further to this conceptual relationship of trust, stake and risk, the authors then define that "trustworthiness" is the objective probability by which the trustee performs a given action on which the welfare of the trustor depends. They explain that well-founded trust is the case in which the "trust" perceived by the trustor equals the "trustworthiness" of the trustee. This means that the trustee possess properties that supports its "trustworthiness" value and hence will not disappoint the trustor when executing the transaction. As "trustworthiness" is defined as an objective

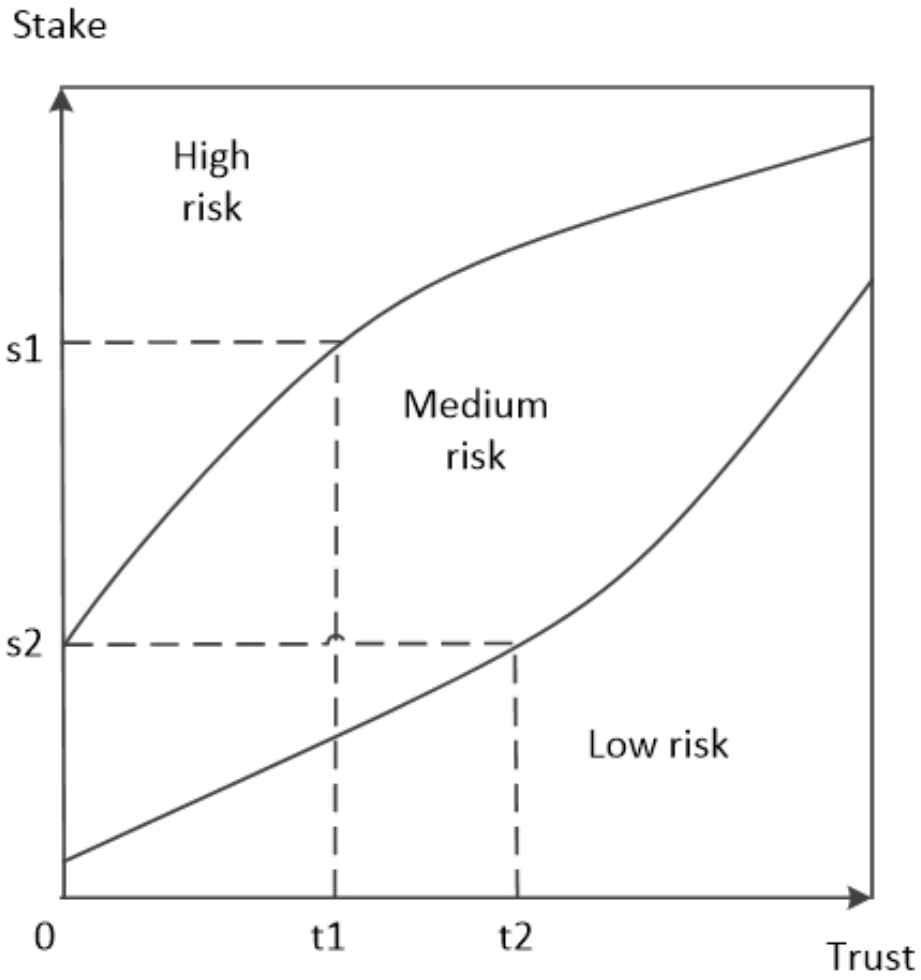


Figure 2.3: Relationship between trust, stake and risk drawn after [82]

variable, we propose that this definition of ”trustworthiness” be inferred as objective trust which was described in the previous section on the user-centered trust model of Hasan et al. This relationship between ”trust” and ”trustworthiness” can be illustrated using the graph in Figure 2.4.

To extend this conceptual relationship to practical application in Information Technology, the parameters for measuring ”trust” (subjective probability that the trustee performs the given task) and the parameters for measuring ”trustworthiness” (objective probability that the trustee performs the given task) have to be defined. However, such measurable parameters are extremely difficult to define [38]. Hence, we limit our investigation to only the high level understanding of this conceptual relationship. Our objective is to find out if there is a connection between different amounts of stake and the level of ”trust” and ”trustworthiness” and not to obtain data to plot the graphs in Figures 2.3 and 2.4. As proposed earlier, ”trust” will be treated as subjective trust while ”trustworthiness” will mean objective trust. Thus, to study this conceptual relationship in Information Technology, the degree of subjective trust will correspond to what the computer users think of the impor-

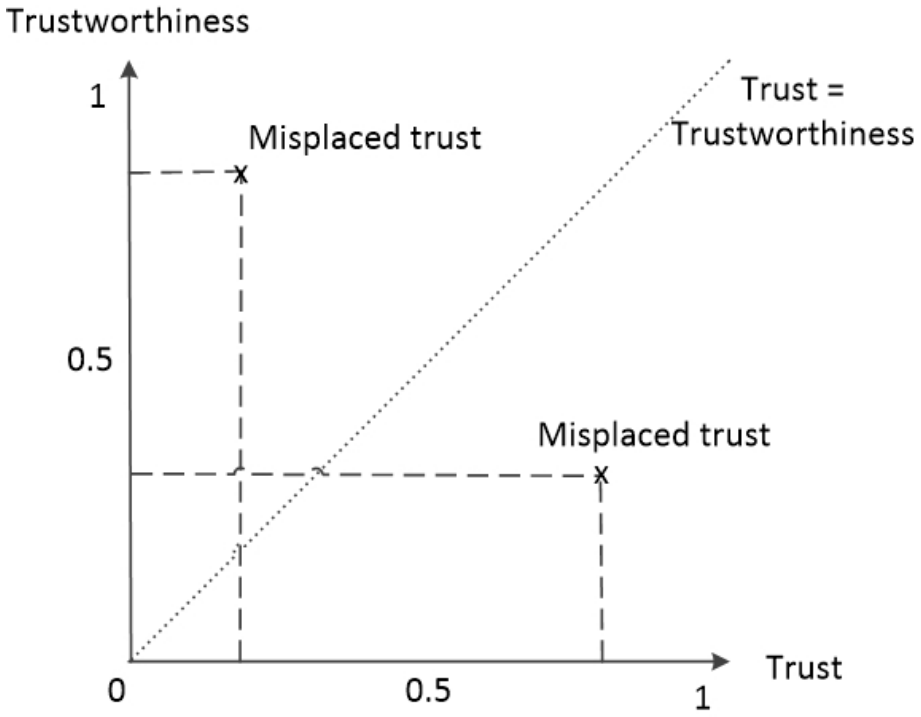


Figure 2.4: Relationship between trust and trustworthiness drawn after [82]

tance of subjective trust notions. This interpretation is extended for objective trust as well. The lists of subjective and objective trust notions have already been captured in Tables 2.1 and 2.2 and we will use these notions in our investigation. We also propose that stake can be interpreted in this study as the value of the information involved in a transaction between the trustor and the trustee in a specific context. The stake can be decided by the trustor, depending on the properties of the computing device and the scenario. Meanwhile, in this study, the risk level is assumed to be fixed. This assumption is inconsequential as the relationship is such that the degree of trust and amount of stake increase in tandem regardless of the risk level.

To summarise Section 2.2, we have reviewed the user-centered trust model of Hasan et al. and captured a list of subjective and objective trust notions. These notions could give rise to attitudes that cause a computer user to form an intention to trust a computing device and eventually carry out the act of using it to perform a specific task. We also reviewed the conceptual relationship from Solhauf et al. of trust, stake and risk where we proposed to associate ”trust” with subjective trust and ”trustworthiness” with objective trust. Stake refers to the value involved when the computer user trusts a computing device to perform a specific task in a certain context. Hence, as explained by the conceptual relationship of Solhauf et al., if risk is fixed, there is a correlation between stake and subjective trust as seen in Figure 2.3 and a correlation between subjective trust and objective trust as seen in Figure 2.4. With this background, we can proceed to form the research questions.

2.3 Research Questions

It is an uphill task to check the practical application of conceptual models because such models are often made with the assumption of ideal conditions. The real world contains many variables and they obfuscate the expected behavior of these models. The task can be made more surmountable by limiting the examination to certain aspects of the concept under selected conditions. Hence, with the understanding of the background to this socio-technical study, the following research questions are proposed.

Question One

What is the degree of importance of the various captured subjective and objective trust notions in influencing the attitude of computer users to the computing device that they use to carry out a given task in a particular scenario?

We understand from [32] and Figure 2.2 that behavior is driven by intentions and intentions are a function of attitudes that are derived from beliefs. We state that, in this study, behavior refers to trusting the computing device. Hence, this research question aims to uncover how computer users think of the importance of the various subjective and objective trust notions when they are forming the attitude that leads to the behavior intention of trusting a computing device to carry out a task in a particular scenario. The idea of studying the degree of importance of the various notions is to gain insight into how these notions influence the attitude of computer users which lead to the behavior of trusting a computing device. The intention is not to prioritize certain technical design requirements.

Question Two

Is there a relationship between stake and the degree of importance given to subjective trust notions, and consequently objective trust notions, when a computer user is considering whether to trust a computing device to carry out a given task in a particular scenario?

Due to the limitation of time and financial resource, we decide to conduct a quick investigation into the practical application, in an Information Technology domain, of one aspect of the conceptual stake, trust and trustworthiness relationship proposed by Solhauf et al.. We interpret in this study that stake is associated with subjective trust and subjective trust is associated with objective trust. Therefore, the question first aims to find out what is the stake that different computer users can accept when using a specific computing device. The result from this inquiry is then linked to how computer users think of the importance of the various subjective and objective trust notions. If the concept is applicable, then the results from the study should indicate a connection between willingness to share information of different value and the degree of importance given to the trust notions. For example, private information may be considered more valuable than public information. Therefore, the notion of data confidentiality may be more important when sharing private information rather than when sharing information already in the public domain.

2.4 Scenarios

To answer the research questions, we developed two scenarios that describe the computer user using some computing devices to carry out the task of information sharing. The objective is for the scenarios to set the context so that we can obtain

data about the acceptable stake for different computer users and how they think of the importance of the subjective and objective trust notions. For the purpose of uncovering the acceptable stake, the scenarios present to the computer users various types of information of different value. The intention is that when computer users select the type(s) of information for sharing, the selection will give an indication of the stake involved. Since this study focuses on user-centered trust notions, we have to assume that any corresponding party is considered trustworthy and the user does not have to take this point into consideration.

2.4.1 Scenario One

The first scenario tells the computer user that he is using a mass market laptop computer to share various types of personal information. The question to ask the computer users and the types of information are listed below. This is an everyday scenario and the majority of computer users will be familiar with it. The types of information listed range from that which can be found in public, such as the information in a telephone directory, to that which is not published online, such as the national identity number. These types of information are considered to be of different value to the computer user due to privacy concerns and the kind of damage that can be done if a malicious actor has access to that information. In other words, they reflect various levels of stake.

Question for Scenario One

Imagine that you are conducting an online information sharing activity using a mass market laptop computer. Please put a tick beside these pieces of information you are willing to share before you decide to find out more about this laptop computer and the sharing protocol, for example, authenticity of digital certificates and cryptographic parameters.

Type of Information for Scenario One

- Information that can be found in the public telephone directory, for example, name, contact number and address.
- Information that can be found on your company's website, for example, workplace address, occupation, designation, email address, work and education history and photo identification.
- Information that can be found on social media, for example, Facebook profile, social activities, list of friends, interest, school attended and personal opinions.
- Information that is unlikely to be published online, for example, national identity number, passport number and bank account number.

2.4.2 Scenario Two

The second scenario is based on the increasingly popular Internet of Things [96]. It narrates that the computer user has several of these embedded computing devices and that they can share a variety of information. The question to ask the computer

users and the types of information are listed below. The types of information range from life style and health data to travel and home environment data. Similar to scenario one, these types of information are considered to be of different value to the computer user. The intention of this scenario is to uncover how a computer user values the different types of information and the importance of the numerous trust notions for Internet of Things.

Question for Scenario Two

Imagine that many embedded computing devices that you use daily are able to communicate with a third party for the purpose of enhancing your quality of life. An example of an embedded computing device is the Smart Refrigerator that can keep track of the amount of food it is storing and order new supplies when the stock runs low. Please put a tick beside these pieces of information that you are comfortable in sharing before you decide to find out more about the embedded computing device and the sharing protocol, for example, authenticity of digital certificates and cryptographic parameters.

Type of Information for Scenario Two

- Information that is related to your life style, for example, diet, movie preference and exercise routine.
- Information that is related to your health, for example, sleeping pattern, pulse rate, blood pressure, body temperature.
- Information that is related to your travel pattern, for example, route, mode of transport, time and meeting schedule.
- Information that is related to your home environment, for example, utilities' consumption and thermostat readings.

2.5 The Survey

A survey was planned as part of the study to examine how a population of computer users views the social and technical aspects of trust in the scenarios described in Section 2.4. We anticipate that the analysis of the survey responses could provide answers to the research questions posed in Section 2.3. The method of an online survey was chosen due to its low cost, convenience and wide reach. A series of test runs were conducted from January to May of 2014. These test runs were conducted in Singapore, United Kingdom and United States of America. The purpose of the test runs was to evaluate the comprehension of the survey questions and identify issues that may affect the quality of the data collected. As a result, the compositions of the survey questions were sharpened to make them easier to understand. In addition, it was found that the survey became difficult to complete if it covered the two scenarios in one go. Hence, the survey was subsequently carried out in two separate and shorter projects, with each project covering one scenario.

The first project covered the scenario of sharing personal information. There were a total of 4 survey questions. The first question asked about the type of information the respondent would be willing to share using a mass market computer

laptop. The question and the list of information are described in Section 2.4.1. The respondent had to choose at least one type of information to share. There were no limit to what the respondent could choose. This was followed by the questions asking the respondent to rank the importance of the subjective and objective trust notions from Tables 2.1 and 2.2. The ordering of the notions in the survey questions was randomized to help improve the quality of the responses. The ranking scale was defined such that a smaller rank value was linked to greater importance of that particular trust notion. The same rank could not be assigned more than one time and all notions had to be ranked. This was to avoid the situation where the respondents banded certain notions together as this could dilute the differentiation of the importance and make analysis difficult. Finally, the survey asked the respondent if they were an early adopter of information technology or only minimally acquired information technology to meet certain needs. The second project covered scenario two and was structured in the same way as the first project. The data obtained from the second project will serve as a comparison to the first project.

The live run of the online survey was conducted on 30 and 31 May 2014. SurveyMonkey [1] was used to deliver the survey questionnaires and collect the responses. The respondents were recruited by SurveyMonkey and their identities were kept anonymous. The target criteria were employed people residing in the United States of America. The collected responses were exported to a spreadsheet and statistical analyses were done using spreadsheet software. The collected responses were analyzed using basic descriptive statistics, followed by hypothesis testing using t-tests in order to assess the correlation between stake, subjective and objective trust notions. The t-test assesses whether the means of two groups are statistically different from each other.

2.6 Analyzing the Results

The respondents were chosen randomly and a total of 156 valid responses were collected for scenario one and a total of 155 valid responses were collected for scenario two. As shown in Table 2.3, the samples are rather balanced with regards to gender. The respondents mostly possess bachelor degrees or higher education qualifications (68.9% and 67.9%) and are well informed about information technology (55.5% and 49.0% are early adopters). Meanwhile, we assume that the collected data is normally distributed.

2.6.1 Scenario One

Figure 2.5 displays the type of information in scenario one and the corresponding percentage of respondents that are willing to share it. The respondents are more willing to share information that can be found in the public domain (62.2% for that which can be found in the public telephone directory) and are less willing to share private and sensitive information (12.2% for that which is unlikely to be published online, for example, national identity number, passport number or bank account number). This result indicates that most of the respondents have a common perception of the value of different types of information and understand the risk of

[1] https://www.surveymonkey.com/

Table 2.3: Demographics of the two samples.

Demographics	Scenario One (n=156)	Scenario Two (n=155)
Female	44.4%	47.8%
Male	55.6%	52.2%
Bachelor Degree and higher	68.9%	67.9%
Associate Degree and High School	31.1%	32.1%
Early adopter	55.5%	49.0%
Minimalist	38.7%	46.5%
Not comfortable with computer technology	5.8%	4.5%

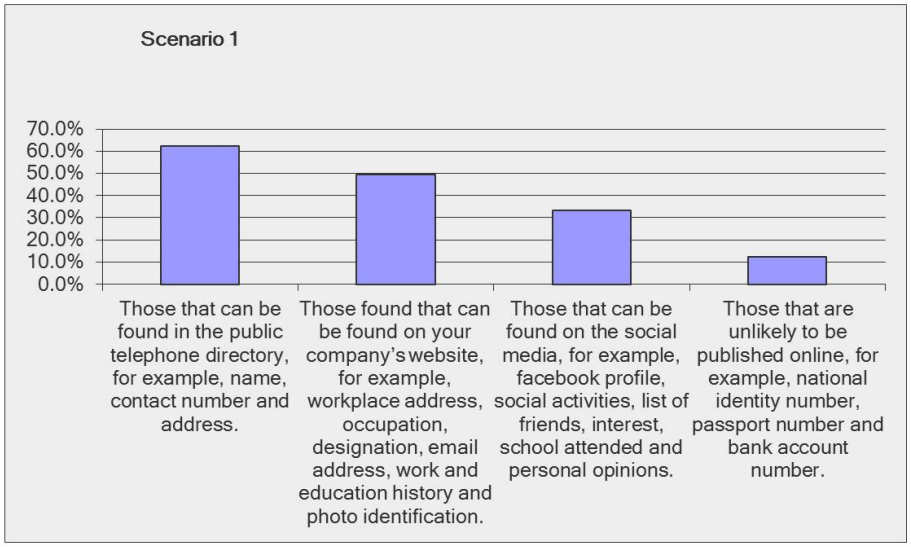


Figure 2.5: Willingness to share different types of personal information in scenario one.

sharing the information online without the computing device meeting certain trust requirements.

Table 2.4 shows the mean (average) rank given by the respondents to the subjective trust notions in scenario one. The result reveals that the respondents place greater value on the vendor's ability to tell the truth and act ethically (mean rank of 2.62). This is followed by the notion that the vendor has the technical competence to solve a problem (mean rank of 3.04). The notion that the vendor is interested in the wellbeing of the respondents has a mean rank of 3.63. Referring back to the user-centered trust model by Hasan et al., this result shows that the notions for trusting beliefs in the vendor plays a more critical role in shaping the respondents' trust in a computing device compared to the notions for the TAM beliefs. The results also validated, in this scenario of human to computer interaction, the application of Mayer et al. [50] and Li et al. [45] proposition that vendor's competence, ability to act ethically and interest in the computer user's wellbeing play a significant role. However, the results suggest that the influence of aesthetics (mean rank 4.99) as

Table 2.4: Ranking of subjective notions in scenario one.

Order	Notion	Mean Rank
1	Vendor's ability to tell you the truth and act ethically when a new issue is discovered.	2.62
2	Vendor's technical competence at solving your problem.	3.04
3	The computing device is useful for this scenario.	3.33
4	The computing device is easy to use.	3.38
5	Vendor's interest in your wellbeing when you encounter a difficulty.	3.63
6	The aesthetics of the computing device suit this scenario.	4.99

proposed by Lindgaard et al. [46] is of less significance compared to other subjective trust notions in this scenario of human to computer interaction. It is noted that the notions of usefulness (mean rank of 3.33) and ease of use (mean rank of 3.38) are ranked closely together and this suggests that the respondents consider them to be equally important.

The ranking of objective trust notions in scenario one is shown in Table 2.5. The result uncovers that the respondents consider information security related notions of confidentiality (mean rank of 2.67) and integrity (mean rank of 3.40) to be very important. Recent high profile news about mass surveillance has heightened the need for privacy [48] and this could have contributed to the observation. The result also shows that notions associated with trusted computing are of lesser importance comparatively. The notion of identity and authenticity of the computing device scores a mean rank of 4.66 while the notion of identity and authenticity of the corresponding party scores a mean rank of 4.76. On the other hand, the notion of expected behavior comes in second last with a mean rank of 5.15. This suggests that the awareness of the benefits of trusted computing such as expected behavior and strong identity could be lacking. It is also noted that the notion of the availability of required function or data (mean rank 5.14) is ranked closely to the notion of expected behavior. This suggests that the respondents do not differentiate between the two in terms of importance. In fact, there is a difference in the benefits brought about by these two notions and an explanation for this finding is that the users lack awareness of the subtleties of these notions. The result also points out that the notion of security standards (mean rank of 4.53) is relatively central to the respondents' trust in the computing device. This observation is not surprising as many vendors have put guides in place to assure the security level of their products and these efforts are included in their marketing material [36]. The notion of origin and change history comes in last with a mean rank of 5.69. This could be because the concept of provenance for trustworthy computing is a new research area and the respondents are not well informed about it. Lastly, it is noted that the notion of identity and authenticity of the mass market laptop computer (mean rank 4.66) and identity and authenticity of the corresponding party (mean rank 4.76) are ranked closely together. This could be that the respondents place equal importance on the

Table 2.5: Ranking of objective notions in scenario one.

Order	Notion	Mean Rank
1	Ensure confidentiality of data.	2.67
2	Ensure integrity of data.	3.40
3	Is certified to meet certain security standard, for example, Common Criteria.	4.53
4	Provides identity and authenticity of the mass market laptop computer.	4.66
5	Provides identity and authenticity of the corresponding party.	4.76
6	Required function or data is always available.	5.14
7	Always work in the same way as before (expected behavior).	5.15
8	Provides the origin and change history of its hardware and software components.	5.69

identity and authenticity of all the parties involved.

A hypothesis test using two tailed independent sample t-test at 5% significance level was carried out to seek an answer to research question two by examining if there is any relationship between stake and subjective and objective trust notions. The 5% significance level is widely used in social science [72].

Hypothesis test 1

H0 : The mean rank of each of the subjective and objective trust notions is equal for the respondents that are willing to share only information found in the public telephone directory and those that are willing to share information found in the public telephone directory, company website and social media.
H1 : The mean rank of each of the subjective and objective trust notions is not equal for the respondents that are willing to share only information found in the public telephone directory and those that are willing to share information found in the public telephone directory, company website and social media.

Analysis. This hypothesis seeks to test if the mean rank of the various notions changes when the stakes are different. There were 47 respondents who are willing to share only information found in the public telephone directory and 24 respondents who are willing to share information found in the public telephone directory, company website and social media. The hypothesis test was carried out for each of the six subjective trust notions and each of the eight objective trust notions. H0 is rejected for the subjective notion of device aesthetics and vendor's technical competency. Specifically, the respondents that are willing to share only information in the public telephone directory are less concerned about aesthetics (mean rank of 5.09 against 4.71) and more concerned about vendor technical competence (mean rank of 2.70 against 3.12). H0 is also rejected for the objective notion of expected behavior. The respondents that are willing to share only information in the public telephone directory place less importance on this notion (mean rank of 5.66 against

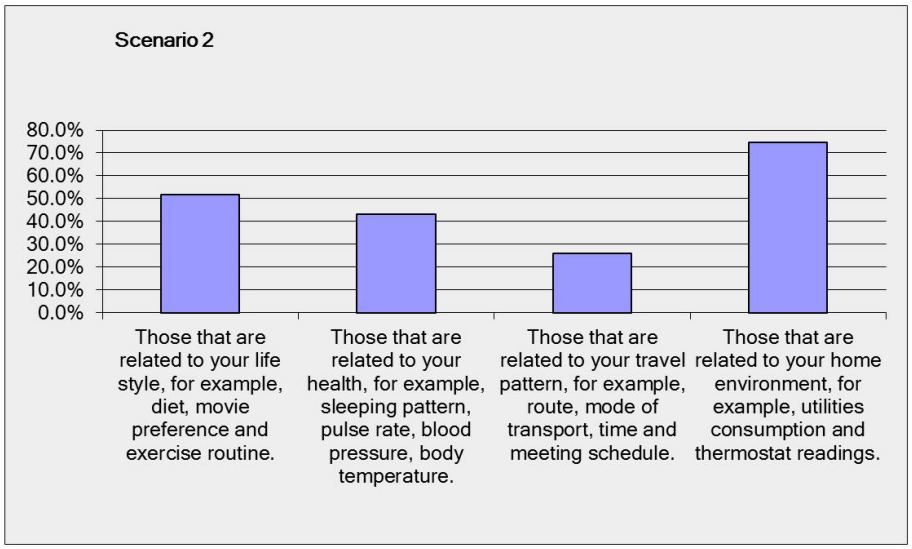


Figure 2.6: Willingness to share different types of personal information in scenario two.

4.79). Further discussion on how this hypothesis testing answers research question two is given in Section 2.7.

2.6.2 Scenario Two

Moving to scenario two, Figure 2.6 presents the type of information and the percentage of respondents that are willing to share this information. It is observed that the respondents are more willing to share information about their home environment (74.8%) and lifestyle (51.6%) for the purpose of improving their quality of life. This result may correspond to the current trend of installing home automation sensors [80] and the popularity of exercise tracking software applications [24]. The finding that 43.2% of the respondents are willing to share information about their health could indicate that more people are ready to embrace the advancements in tele-medicine. On the other hand, the respondents are less willing to share their travel pattern (25.8%). This result suggests that the respondents are uncomfortable with sharing their geo-location data and calendar although these functions are already widely available in computing devices such as the smart phone. Researchers and vendors of Internet of Things computing devices will benefit from this collected data as it provides a view of how the respondents value the different types of information that could be shared by this means.

The ranking of subjective trust notions is shown in Table 2.6. Although the notion of vendor telling the truth and acting ethically (mean rank 2.62) is still the most important to the respondents, the notion that the computing device is easy to use is now ranked second with a mean rank of 3.01. This suggests that the respondents place significant emphasis on the ease of use of these Internet of Things devices. The ranking for the remaining notions is not dissimilar to that in scenario one.

The ranking of objective trust notions is shown in Table 2.7. The rank of the notions is largely similar to those in scenario one. This implies that the requirement

Table 2.6: Ranking of subjective notions in scenario two.

Order	Notion	Mean Rank
1	Vendor's ability to tell you the truth and act ethically when a new issue is discovered.	2.62
2	The computing device is easy to use.	3.01
3	Vendor's technical competence at solving your problem.	3.25
4	The computing device is useful for this scenario.	3.30
5	Vendor's interest in your wellbeing when you encounter a difficulty.	3.83
6	The aesthetics of the computing device suit this scenario.	5.00

Table 2.7: Ranking of objective notions in scenario two.

Order	Notion	Mean Rank
1	Ensure confidentiality of data.	2.06
2	Ensure integrity of data.	3.63
3	Is certified to meet certain security standard, for example, Common Criteria.	4.33
4	Provides identity and authenticity of the corresponding party.	4.65
5	Provides identity and authenticity of embedded computing device.	4.69
6	Always work in the same way as before (expected behavior).	5.12
7	Required function or data is always available.	5.25
8	Provides the origin and change history of its hardware and software components.	6.27

for trust in embedded computing devices is the same as those for a mass market laptop computer which has a more complex build, diverse functionality and handles more types of information. Once again, it is observed that the ranking for the notion of identity and authenticity of the embedded computing device (mean rank 4.69) and identity and authenticity of the corresponding party (mean rank 4.65) are ranked closely together. This finding confirms that the respondents do not differentiate between these two notions and they place equal importance on the identity and authenticity of all the parties involved. Close ranking is also repeated for the notion of expected behavior (mean rank 5.12) and the notion of availability of required function or data (mean rank 5.25). This confirms the earlier inference that more awareness may be required for the core concept of expected behavior in trusted computing.

Hypothesis testing using two tailed independent sample t-test at 5% significance level was carried out to seek an answer to research question two by exam-

ining if there is any relationship between stake and subjective and objective trust notions.

Hypothesis test 2

H0 : The mean rank of each of the subjective and objective trust notions is equal for the respondents that are willing to share only information related to home environment and those that are willing to share information related to lifestyle, health, travel pattern and home environment.
H1 : The mean rank of each of the subjective and objective trust notions is not equal for the respondents that are willing to share only information related to home environment and those that are willing to share information related to lifestyle, health, travel pattern and home environment.

Analysis. This hypothesis seeks to test if the mean rank of the various notions changes when the stakes are different. There were 36 respondents who are willing to share only information related to home environment and 21 respondents who are willing to share information related to lifestyle, health, travel pattern and home environment. Similar to hypothesis test 1, this test was carried out for each of the six subjective trust notions and eight objective trust notions. H0 is rejected for the subjective notion of device usefulness, vendor's interest in respondent's wellbeing, vendor's ability to tell the truth and act ethically and vendor's technical competency. Specifically, the respondents that are willing to only share information related to home environment are less concerned about vendor's interest in the respondent's wellbeing (mean rank of 4.14 against 3.62) and vendor's ability to tell the truth and act ethically (mean rank of 2.94 against 2.26) and more concerned about the vendor's technical competence (mean rank of 3.00 against 3.76). H0 is also rejected for the objective notion of expected behavior and confidentiality of data. The respondents who are willing to share only information related to home environment place more importance on expected behavior (mean rank of 4.81 against 5.57) and less importance on data confidentiality. Further discussion on how this hypothesis testing answers research question two is given in Section 2.7.

2.7 Discussion

The data collected from the responses to the online survey has answered the research question of which subjective and objective trust notions are considered to be important to influencing computer users' attitudes towards trusting a computing device to carry out a given task in a particular scenario. The ranking of the various subjective and objective trust notions in the two scenarios are shown in Tables 2.4, 2.5, 2.6 and 2.7. The results uncover that the trusting beliefs in vendors described in the conceptual user centered trust model play an important role because this notion is always ranked higher. The implication of this finding is that computer users are likely to trust computing devices from vendors who are reputed to have good technical competence and have an excellent record of acting truthfully and ethically. However, these findings may have an ominous application if a malicious entity who wishes to compromise the computer user makes use of these findings to improve its social engineering tactics. For example, a computer user may be

tricked into trusting a malicious application which appears to be originating from a reputable vendor.

On the topic of objective trust notions, the results show that the information security attributes of confidentiality and integrity are ranked highly. The results also show that technical attributes associated with trusted computing technology such as the TPM are of lesser importance compared to information security attributes. Nevertheless, the survey provides information for the development of TPM-based consumer use cases. For example, as the responses indicated that information security attributes are important, consumer use cases could focus on using the TPM's cryptographic engine to provide confidentiality and integrity protection for data. Meanwhile, TPM vendors could build an image that portrays ethical behavior and technical competence so as to positively influence the beliefs of computer users and consequently their attitudes towards this technology.

On the research question of whether there is a relationship between stake and degree of importance of subjective trust and objective trust, the survey found that computer users consider the importance of the various subjective and objective trust notions differently when the stakes are not the same. It is seen from hypothesis test 1 that the respondents who are willing to share only information in the public telephone directory are less concerned about aesthetics and more concerned with the vendor's technical competence. It could be that this group of respondents is acutely aware of the risk of online information sharing and to mitigate this risk, they require the vendor of the mass market laptop computer to have competent technical knowledge. H0 of hypothesis test 1 is also rejected for the objective notion of expected behavior. The respondents that are willing to share only information in the public telephone directory place less importance on this notion. This could indicate a shift in the ranking of other objective trust notions but the change is not detected by the statistical test.

We can see a trend forming when we look at hypothesis test 2. Respondents who are only willing to share information related to home environment place more importance on vendor's technical competence. This is similar to the observation in hypothesis test 1 for respondents who are only willing to share information in the public telephone directory. We also noted that the statistical difference is more marked in scenario two as hypothesis test 2 uncovers changes in importance to the subjective trust notions of device usefulness, vendor's interest in respondent's wellbeing and vendor's ability to tell the trust and act ethically. Meanwhile, H0 is rejected for the objective notions of expected behavior and confidentiality of data in hypothesis test 2. We see that respondents who only share information related to home environment place more importance on expected behavior and less importance on data confidentiality. However, compared to scenario one, this observation is not aligned. For example, respondents who only share information found in the public telephone directory in scenario one place less importance on expected behavior and this is the opposite for respondents who only share information related to home environment in scenario two as they place more importance on the same notion. This observation indicates that the correlation between subjective trust and objective trust as shown in Figure 2.4 is not detected across different scenarios although there is a shift in the ranking of objective trust notions within a scenario when the stake changes.

Overall, when we cross reference the results from hypothesis tests 1 and 2 to the conceptual relationship of trust, stake and risk proposed by Solhauf et al. [82], we can see that as the stake increases from sharing less information to sharing more information, there are changes to the importance of the various subjective and objective trust notions within each scenario but only the subjective trust notion of vendor's technical competence shows a consistent pattern across scenarios.

2.8 Summary

In this chapter, a socio-technical study on user centered trust notions and their association to stake for practical Information Technology scenarios is presented. This socio-technical study made use of an online survey to investigate how computer users perceive subjective and objective trust notions and their association to the value of information involved in the transaction. The online survey provided data on how the respondents rank the importance of the subjective and objective trust notions in the two scenarios. This data helped to answer the research question on the degree of importance of the subjective and objective trust notions in influencing the attitude of the computer users towards a computing device that they use to carry out a task in a particular scenario.

The study also attempted a quick investigation into the practical application of the conceptual relationship between stake, subjective trust and objective trust. Hypothesis testing on the survey data was carried out to examine how one group of respondents who are willing to share more information rank the various trust notions compared to another group of respondents who are less willing to share information. Within each scenario, we detected changes to the perceived importance of certain subjective and objective trust notions when the stake changed but there was no consistent pattern other than that the subjective trust notion of the vendor's technical competence was ranked consistently higher when the stake was higher in both scenarios.

Thus, this study contributes towards the current understanding of how computer users view the social and technical aspect of trust in practical scenarios. The cross disciplinary research also provides an insight into the connection between stake and computer users' perceptions of trust notions. Most importantly, this study establishes the understanding of trust notions for this thesis.

Chapter 3

A Causality-based Model for Describing the Trustworthiness of a Computing Device

3.1 Introduction

We understand from Chapter 1 that a key component to building end-to-end trust is the ability to describe the trustworthiness of a computing device. There are two facets to this ability. First of all, it is the description of the capabilities of a computing device that could give rise to its trustworthiness. This requires marking up this description with metadata in a well understood and consistent manner. Such annotation will enable the description of the capabilities to be processed by a computer [41]. In this chapter, we take the first step towards describing the capabilities of a computing device. Meanwhile, the second facet refers to evidence that conveys assurance in certain capabilities of the computing device. A prevailing technique is the use of attestation which vouches for the identity and state of a computing device's software stack. In Chapter 4, we propose to use provenance as evidence during attestation .

This novel causality-based model for describing the capabilities of a computing device that give rise to its trustworthiness contributes to Grawrock's research agenda on a trust language that could describe the trustworthiness of a computing device[1]. As far as we know, this is the first attempt at this challenge. The main feature of this approach is that the description captures the causal dependencies between trust notions, capabilities, mechanisms and configurations, and this information is useful for intelligent processing. Moreover, it uses a graph to describe a computing device at various levels of abstraction in a way that is easily comprehended. Lastly, the causal graph can be made machine-readable by translating it into a format that uses markup language, such as the eXtensible Markup Language (XML).

The rest of this chapter is organized as follows. Section 3.2 gives the background to the causality-based approach. Section 3.3 defines the concept of causality for this model and makes clear the semantics of the terms used. Section 3.4 defines the graph and illustrates how it is used to show the dependencies between trust notions, capabilities, computing mechanisms and their configuration. This is followed by Section 3.5, which describes how the graph can be implemented as an XML schema and we discuss how this schema was applied to the Metadata Access Point (MAP) database server of the Trusted Network Connect (TNC) open architecture. Section 3.6 explains how we can carry out trust assessment. Section 3.7

[1] http://www.iaik.tugraz.at/content/about_iaik/events/ETISS_INTRUST_2013/ETISS/slides/DavidGrawrock.pdf

reviews related works. The summary to this chapter is in Section 3.8.

3.2 A Causality-based Approach

The design of the model aims to meet the following requirements:

- To describe the capabilities of a computing device that could give rise to its trustworthiness.
- To define the model in a clear and easy to understand manner.
- To support the digital representation of this model by translating it into a machine-readable format.

The intention of these requirements is to guide the development of the model and make sure that it can be implemented in practice. We do not intend to specify how the descriptions are created and updated but we can assume that such descriptions are created by the designer of the computing device and the descriptions are updated by agents installed on that computing device. While we refer to the objective trust notions described in Chapter 2 as we develop this model, we exclude subjective trust notions because their perception can differ from person to person.

To describe the capabilities of a computing device that give rise to its trustworthiness, we refer to the technical models described in the National Institute of Standards and Technology (NIST) Special Publication 800-33 [84]. These models encompass information at several levels. They range from high level security notions to low level specific technical details. For example, the low level technical mechanism of cryptographic key management is described as enabling the capability of access control enforcement which in turn supports the security notion of confidentiality. The NIST publication is intended to describe the technical foundations that underlie security capabilities. Since our intentions are broadly aligned, we will frame our description of the capabilities of a computing device using the structure of the technical models described in the NIST publication. The description will cover the trust notions, capabilities, mechanisms and configurations of a computing device. We also note that if any of the technical foundations are missing, the resulting high level capability and notion will not exist. We interpret this observation as causality.

A general causal model consists of a set of equations of the form

$$x_i = f_i(pa_i, u_i),\ i = 1,...,n,$$

where pa_i stands for the set of variables that directly determine the value of x_i and where u_i represents errors or disturbances due to omitted factors [63]. This functional relationship can be thought of as Laplace's quasi-deterministic concept of causality. Bayesian networks are usually used to represent this general causal model and, for example, u_i can be used to indicate the probability of a causal dependency when affected by factors such as an attack on the computing device. However, this approach is sophisticated and at this stage, it is beyond our design requirements. Hence, we decided to set u_i to zero (i.e. no errors) in our causality-based model. By setting u_i to zero, the causal model loses the ability to deal with the

probability of a causal dependency. As our primary concern is to introduce a model to describe the causal dependencies between trust notions, capabilities, computing mechanisms and their configuration and to make sure that this model works in practice, omitting the above ability does not affect the description. Nevertheless, we will discuss more about setting u_i to non-zero in Chapter 8.

With this understanding of a causal model, we can say that: *A cause is defined to be an object followed by another, where, if the first object had not been, the second would never had existed* [44]. This concept of causality has important applications in computer science, such as intelligent planning and processing [30]. Whenever we seek to explain a set of computations that unfold in a specific scenario, the explanation produced must address the cause and effect of these computations. The generation of such explanations requires the analysis of the concept of causality and the development of a data model to characterise the account.

On the other hand, the practical application of causality requires it to transform into a graph that is founded on mathematics and logic [63]. A directed graph consists of a set V of vertices and a set E of edges. The vertices in this graph represent the variables and the edges denote a certain causal relationship holds between pairs of variables. As a causal explanation describes how a variable is caused by another variable, this description of dependency path can be written as a triplet (v1, e, v2) where v1, v2 $\in V$ and e $\in E$. In other words, a source variable v1 is related to destination variable v2 through the causal dependency e. Consequently, this can also be expressed as a graph where e is an edge from source vertex v1 to destination vertex v2. Hence, we can conjecture that a causal graph composes of multiple triplets and they form a larger graph with numerous vertices that are connected by edges.

With this background knowledge, we can envisage that this graph will have a data structure that depicts the causal dependences between the low level technical mechanisms and high level capabilities and notions. Although it can be argued that such information can be obtained from various sources, the main advantage of this model is that the information about technical mechanisms, configuration, capabilities and trust notions are linked together meaningfully by their causal dependencies. As a result, another party can carry out intelligent processing on this information and decide how it will interact with this computing device.

3.3 Basic Definitions

In this section, we define the semantic meaning of the terms used in this causality-based model. As the causality-based model has to be transformed into a graph for practical application, Definitions 2 to 5 refer to the kind of vertices in the graph.

Definition 1 (Causality) We refer to the definition of causality in the previous section and define causality in this model to be the use of a configured mechanism, or a set of configured mechanisms, and if one of the configured mechanisms is not used, the capability and the resulting trust notion that arise out of the use of the configured mechanisms will not exist. For example, the trust notion of confidentiality relies on the capability of disk encryption which in turn is derived from the usage of a symmetric cryptographic mechanism.

We then produce the following definitions for the key terms in definition 1.

Definition 2 (Mechanism) It is a computation process that has at least one input and at least one output. Although how the computation works is defined by computer code, the quality of the output can be influenced by the applied configuration. For example, a symmetric encryption mechanism is configured to use a symmetric key of certain size, then takes in data and produces the encrypted form of that data.

Definition 3 (Configuration) A set of parameters that affect the output of a mechanism. For example, the key size of an AES symmetric encryption mechanism needs to be specified as different applications require different key sizes.

Definition 4 (Trust Notion) A notion reflects a specific behavior of a computing system. We understand that there is a large body of research on trust notions. Thus, to focus our research effort, we scoped this work to the trust notions described in Chapter 2. These trust notions are confidentiality, integrity, identity, authenticity, availability and expected behaviour. These are objective notions that are related to technical properties of a computing device.

Definition 5 (Capability) A capability can be considered as a high level description of a mechanism or a set of cooperating mechanisms. It refers to the ability to perform certain tasks. A capability of one computing system is the same as another system if this capability is derived from the same set of mechanisms and configurations.

Various notions of causal dependencies were considered for the causality-based model. A strong notion of causal dependency would provide a detailed explanation of how an effect is caused. However, such a strong notion of causal dependency was not practical as one could argue that additional factors may have influenced the outcome. For example, if the computation has been occurring on a hardware that is operating within its allowed temperature range. Therefore, we decided that weaker notions which describe only the core meaning of the causal dependencies would be more suitable. This decision is supported by two considerations:

- *Usability*. We expect that the description of the capabilities could be produced without a detailed knowledge of how the mechanisms interact and how the configuration affects the behaviour of the mechanisms. Thus, weaker notions allow the causality-based model to be used in practice by non experts.

- *Composability*. We desire that multiple causal graphs can be combined to reflect more complex capabilities. This is true in practical applications whereby a complex capability can be composed of various mechanisms spread across diverse computing systems. Hence, weaker notions allow a more flexible interpretation of the causal dependencies and avoid complications due to contrasting explanation used by different computing systems.

Nevertheless, stronger notions of causality for specific applications, can be developed as subclasses to the dependencies defined in our model. While the above definitions refer to the kind of vertices in a causal graph, the edges between the nodes will represent their causal dependencies. On this point, we propose the following causal dependencies for this model:

Definition 6 (Mechanism *CallsOn* Mechanism) A mechanism can call on another mechanism during its computation process. This relation can be one to one,

one to many, many to one or many to many. However, this causal dependency is affected by the configuration of the mechanism. A mechanism with the same configuration can be called on multiple times by other mechanisms. If this mechanism has another configuration, then it must be represented again on the causal graph. A calling mechanism can only complete its computation process when the mechanism that it called on has completed its computation process. For example, a software application calls on a key generator and a symmetric encryption mechanism when it is performing disk encryption.

Definition 7 (Capability *DerivesFrom* Mechanism) A capability is derived from a mechanism or a set of cooperating mechanisms. If a capability is derived from a set of cooperating mechanisms, all these mechanisms must have begun and completed their computation processes before the capability can exist. The same constraint applies to the situation when a capability is derived from one mechanism. The quality of a capability will be affected if a mechanism or a configuration is not from a specified set. For example, the capability of disk encryption is derived from a software application that encrypts data using symmetric cryptography and manages the key used in the encryption process.

Definition 8 (Trust Notion *ReliesOn* Capability) The existence of a trust notion relies on a capability or a set of capabilities. The mechanisms that the capability depends on must have completed their computation process before the trust notion can exist. A trust notion can rely on more than one capability. For example, the trust notion of confidentiality relies on the capability of disk encryption. The same trust notion of confidentiality can also rely on the capability of network encryption which uses a different set of mechanisms from the capability of disk encryption.

Definition 9 (Mechanism *Uses* Configuration) Each mechanism has a maximum of one configuration for each account of causality. In other words, it is a one to one relation. If the same mechanism uses more than one configuration, then the mechanism shall be represented again with another configuration.

We have introduced the definition of the terms used in the causality-based model. However, there may be ambiguity in the definition and hence we introduce set-theoretic definitions in Figure 3.1 to further clarify the causality-based model.

In Figure 3.1, lines 1 to 4 define the term mechanism, configuration, trust notion and capability as sets. These terms correspond to vertices when transformed to a graph. Line 5 further defines that C is a subset of CAPABILITY and element c only belongs to C if and only if c has a property P. P refers to the property that elements of C work together to give rise to a trust notion. Line 6 defines that M is a subset of MECHANISM and element m only belongs to M if and only if m has a property Q. Q refers to the property that elements of M work together to give rise to a capability. Line 7 says that CF is a subset of CONFIGURATION and that element cf belongs to CF if and only if cf has a property S. S refers to the property that elements of CF are applicable to a particular set of mechanisms that give rise to a capability.

Lines 8 and 9 are the interpretation of the definition of causality in this model. It says that there is a function f such that it maps elements of the TRUSTNOTION to the set CAPABILITY. Then there is a complex function g such that g(C) produces a set that contains the mappings of M to CF. Lines 10 to 13 refer to the edges that link the vertices. *CallsOn*, *ReliesOn* and *DerivesFrom* refers to a one to one or one to many mapping. For *Uses*, we use the symbol for an partial injective function

1. [MECHANISM] the set of all possible mechanisms in computing systems.
2. [CONFIGURATION] the set of all possible configurations in computing systems.
3. [TRUSTNOTION] the set of all possible trust notions enabled by computing systems.
4. [CAPABILITY] the set of all possible capabilities provided by computing systems.

5. C is a subset of CAPABILITY such that the elements of C are capabilities that work together to give rise to a trust notion. Therefore,
   C == { c:CAPABILITY | P(c) }
6. M is a subset of MECHANISM such that the elements of M are mechanisms that work together to give rise to a capability. Therefore,
   M == { m:MECHANISM | Q(m) }
7. CF is a subset of CONFIGURATION and elements of CF are configurations for a particular set of mechanisms. Therefore,
   CF == { cf:CONFIGURATION | S(cf) }

   Following definition 1, we have:

8. f : TRUSTNOTION $\rightarrow$ CAPABILITY
9. g : C $\rightarrow$ (M $\rightarrowtail$ CF)

10. CallsOn : MECHANISM $\rightarrow$ MECHANISM
11. ReliesOn : TRUSTNOTION $\rightarrow$ CAPABILITY
12. Uses : MECHANISM $\rightarrowtail$ CONFIGURATION
13. DerivesFrom : CAPABILITY $\rightarrow$ MECHANISM
14. V: TRUSTNOTION $\rightarrow$ MECHANISM
    W: TRUSTNOTION $\rightarrow$ CONFIGURATION
    X: CAPABILITY $\rightarrow$ CONFIGURATION
15. $\forall$ tn:TRUSTNOTION, m:MECHANISM, (tn,m) $\in$ V | $\nexists$ v:V $\bullet$ tn V m
16. $\forall$ tn:TRUSTNOTION, cf:CONFIGURATION, (tn,cf) $\in$ W | $\nexists$ w:W $\bullet$ tn W cf
17. $\forall$ cp:CAPABILITY, cf:CONFIGURATION, (cp,cf) $\in$ X | $\nexists$ x:X $\bullet$ cp X cf

Figure 3.1: Definition of causality-based model.

to say that each mechanism has only one configuration at a time and there may exist a mechanism that does not need any configuration. These edges shall be directed and the direction is from the range to the domain of the functions CallsOn, ReliesOn, DerivesFrom and Uses. The way these edges are directed reflects our requirement to describe the causal dependencies between trust notions, capabilities, mechanisms and configurations. This is a mapping from abstract concepts to low level technical primitives. Meanwhile, the directed edge ReliesOn is the manifestation of function f in line 8 while the directed edge DerivesFrom is the manifestation of function g in line 9. Finally, lines 14 to 17 clarify that composition is not allowed. Particularly, it addresses the constraint that neither trust notion nor capability can arise only out of configuration. The other reason is that we want to capture all the causal dependencies.

3.4 Graph Definition

This section defines the graph that is required for the practical application of the causality-based model. We name this as the causal graph. The following defines this causal graph.

1. Vertices of the causal graph are the elements of MECHANISM, CONFIGURATION, CAPABILITY and TRUSTNOTION and they are given unique identifiers. Two elements from the same set are equivalent if they have the same identifiers.

2. Elements of MECHANISM, CONFIGURATION and CAPABILITY can be assigned to a computing system. It means that these elements are hosted by a particular computing system. This supports composability; i.e a capability can be derived from mechanisms held in more than one computing system. This also supports the situation where multiple capabilities from different systems give rise to a trust notion.

3. Edges of the causal graph are identified by the vertices they connect. Vertices are elements of the sets MECHANISM, CONFIGURATION, CAPABILITY and TRUSTNOTION and they have unique identifiers.

4. A causal graph is a set of vertices and edges as specified in this chapter.

5. A proper causal graph contains a ReliesOn and a DerivesFrom edge. In other words, it explains the existence of a trust notion and capability. This ensures that a causal graph captures a valid causal dependency between the high level trust notion or capability and a low level technical mechanism. A proper causal graph is also acyclic due to the direction of the edges. It is possible that with the additional definition of edges, a causal graph can be made cyclic but this will not be discussed in this chapter.

Figure 2 provides a set-theoretic definition of the causal graph. Lines 1 to 5 declare the sets of identifiers. Lines 6 to 8 say that the vertices Mechanism, Configuration and Capability have unique identifiers and are associated with at least one

1. MechanismId : primitive set
2. ConfigurationId : primitive set
3. TrustNotionId : primitive set
4. CapabilityId : primitive set
5. System : primitive set
6. Mechanism : MechanismId $\rightarrow \mathbb{P}$(System)
7. Configuration : ConfigurationId $\rightarrow \mathbb{P}$(System)
8. Capability : CapabilityId $\rightarrow \mathbb{P}$(System)
9. TrustNotion : TrustNotionId
10. CallsOn $\subseteq$ Mechanism $\times$ Mechanism
11. ReliesOn $\subseteq$ TrustNotion $\times$ Capability
12. Uses $\subseteq$ Mechanism $\times$ Configuration
13. DerivesFrom $\subseteq$ Capability $\times$ Mechanism
14. CausalGraph $\subseteq \mathbb{P}$(CallsOn) $\times \mathbb{P}$(ReliesOn) $\times \mathbb{P}$(Uses) $\times \mathbb{P}$(DerivesFrom)
15. $\forall$ m1,m2 $\in$ Mechanism, cf1, cf2 $\in$ Configuration, s1, s2 $\in$ System.
    $\exists$ u1, u2 $\in$ Uses | u1 = ((m1,s1),(cf1, s1)), u2 = ((m2, s2), (cf2, s2)) $\bullet$ u1 = u2 $\Leftrightarrow$ (m1 = m2) $\wedge$ (cf1 = cf2) $\wedge$ ((s1 = s1) $\vee$ (s1 $\neq$ s2))
16. $\forall$ m1,m2, m3, m4 $\in$ Mechanism, u1, u2, u3, u4 $\in$ Uses, cf1, cf2, cf3, cf4 $\in$ Configuration, s1, s2 $\in$ System.
    $\exists$ co1,co2 $\in$ CallsOn | co1 = ((m1,s1),(m2,s1)), co2 = ((m3,s2), (m4, s2)), u1 = ((m1, s1),(cf1, s1)), u2 = ((m2, s1), (cf2, s1)), u3 = ((m3, s2),(cf3, s2)), u4 = ((m4, s2), (cf4, s2)) $\bullet$ co1 = co2 $\Leftrightarrow$ (m1 = m3) $\wedge$ (m2 = m4) $\wedge$ (cf1 = cf3) $\wedge$ (cf2 = cf4) $\wedge$ ((s1 = s1) $\vee$ (s1 $\neq$ s2))
17. $\forall$ DF1, DF2 $\subseteq$ DerivesFrom, M10, M11, M20, M21 $\subseteq$ Mechanism, CO1, CO2 $\subseteq$ CallsOn, U10, U11, U20, U21 $\subseteq$ Uses, CF10, CF11, CF20, CF21 $\subseteq$ Configuration.
    $\exists$ cp1, cp2 $\in$ Capability | DF1 = (cp1, M10), DF2 = (cp2, M20), CO1 = (M10, M11), CO2 = (M20, M21), U10 = (M10, CF10), U20 = (M20, CF20), U11 = (M11, CF11), U21 = (M21, CF21) $\bullet$ cp1 = cp2 $\Leftrightarrow$ (U10 = U20) $\wedge$ (U11 = U21) $\wedge$ (CO1 = CO2)
18. $\forall$ RO1, RO2 $\subseteq$ ReliesOn, CP1, CP2 $\subseteq$ Capability, DF1, DF2 $\subseteq$ DerivesFrom, M10, M11, M20, M21 $\subseteq$ Mechanism, CO1, CO2 $\subseteq$ CallsOn, U10, U11, U20, U21 $\subseteq$ Uses, CF10, CF11, CF20, CF21 $\subseteq$ Configuration.
    $\exists$ tn1, tn2 $\in$ TrustNotion | RO1 = (tn1, CP1), RO2 = (tn2, CP2), DF1 = (CP1, M10), DF2 = (CP2, M20), CO1 = (M10, M11), CO2 = (M20, M21), U10 = (M10, CF10), U20 = (M20, CF20), U11 = (M11, CF11), U21 = (M21, CF21) $\bullet$ tn1 = tn2 $\Leftrightarrow$ (DF1 = DF2) $\wedge$ (CO1 = CO2) $\wedge$ (U10 = U20) $\wedge$ (U11 = U21)

Figure 3.2: Defining the causal graph.

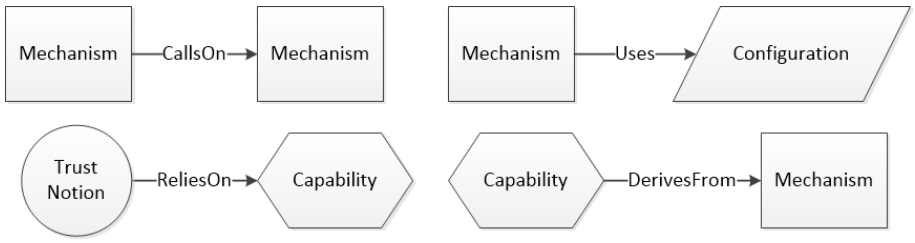


Figure 3.3: Pictorial representation of the vertices and edges in the causal graph model.

computing system. In Line 9, we do not link Trust Notion to a particular computing system as it refers to a universal quality. The set for edges CallsOn, ReliesOn, Uses and DerivesFrom are specified from line 10 to 13. The information about the vertices and edges will be expressed as a triplet (v1, e, v2) where vertex v1 is related to vertex v2 through a causal dependency e. Line 14 says that a causal graph is a set containing the triples that describe the causal dependencies.

Lines 15 to 18 attempt to explain systematically how two graphs can be equal. Line 15 looks at the most basic causal dependency between a mechanism and a configuration. It says that their Uses edges are the same if the corresponding mechanisms and configurations are the same although their systems may not be the same. We then proceed to line 16 that defines how two CallsOn edges are the same. It says that the CallsOn edges between two separate sets of mechanisms are the same if the elemental mechanisms from each set are the same and they use the same configurations although their systems are not the same. Lines 17 and 18 make use of the definitions in lines 15 and 16. They explain how two capabilities or two trust notions can be the same. Line 17 says that two capabilities can be the same if they have the same set of Uses and the same set of CallsOn. In other words, two capabilities are the same if they are derived from the same set of mechanisms and these set of mechanisms have the same set of configurations. Line 18 says that two trust notions can be the same if they rely on the same capabilities, and these capabilities are derived from the same mechanisms, and these mechanisms use the same configurations. Note that the requirement that the system may or may not be the same is implicit.

At this point, we will introduce the basic pictorial representation of vertices and directed edges of the causal graph. Mechanisms are represented as rectangles while configurations are trapeziums. Finally, trust notions are circles and capabilities are hexagons. Figure 3.3 shows this pictorial representation.

3.5 Praxis

We applied the causality-based model to the Metadata Access Point (MAP) server of the Trusted Network Connect (TNC) open architecture. TNC is specified by the Trusted Computing Group and it enables the application and enforcement of security requirements for endpoints connecting to an enterprise network [89]. The MAP server acts as a database where metadata that describes endpoints is published. Security devices can subscribe to the MAP server and read the published

metadata as part of a security process such as access control. Communications with the MAP server are carried out over the Interface for Metadata Access Point (IF-MAP) protocol. Standard sets of metadata are defined for use by the MAP server to determine information about an endpoint such as device status, location and characteristics. There are currently two standards for metadata; the IF-MAP Metadata for Network Security [90] and IF-MAP Metadata for Industrial Control System [91]. The specification of the standards defines the structure and content of the metadata and includes XML schemas to represent this information. We created a new XML schema to represent our causality-based model. This new XML schema is presented in Listing 3.1. This new XML schema declares the four types of edges as complex elements. Within the complex element, the vertices are further declared as complex elements. Each complex element contains additional elements such as id and system. On the declaration for the directed edge CallsOn, we used the term ”MainMechanism” to represent the calling mechanism and the term ”SubMechanism” to represent the mechanism that is being called on. This avoids the confusion caused when the term ”Mechanism” is not differentiated.

```
1  <?xml version=”1.0”?>
2  <xsd:schema targetNamespace=”causal_graph”
3  xmlns:xsd=”http://www.w3.org/2001/XMLSchema”>
4  <xsd:element name=”causal_graph_data”>
5  <xsd:complexType>
6  <xsd:choice maxOccurs=”unbounded”>
7  <xsd:element name=”causal_graph_id” type=”xsd:string”/>
8
9  <!−−CallsOn−−>
10 <xsd:element name=”CallsOn”>
11 <xsd:complexType>
12 <xsd:sequence>
13 <xsd:element name=”MainMechanism”>
14 <xsd:complexType>
15 <xsd:sequence>
16 <xsd:element name=”id” type=”xsd:string”/>
17 <xsd:element name=”system” type=”xsd:string”/>
18 </xsd:sequence>
19 </xsd:complexType>
20 </xsd:element>
21 <xsd:element name=”SubMechanism” maxOccurs=”unbounded”>
22 <xsd:complexType>
23 <xsd:sequence>
24 <xsd:element name=”id” type=”xsd:string”/>
25 <xsd:element name=”system” type=”xsd:string”/>
26 </xsd:sequence>
27 </xsd:complexType>
28 </xsd:element>
29 </xsd:sequence>
30 </xsd:complexType>
31 </xsd:element>
```

```
32
33 <!--ReliesOn-->
34 <xsd:element name="ReliesOn">
35 <xsd:complexType>
36 <xsd:sequence>
37 <xsd:element name="TrustNotion">
38 <xsd:complexType>
39 <xsd:sequence>
40 <xsd:element name="id" type="xsd:string"/>
41 </xsd:sequence>
42 </xsd:complexType>
43 </xsd:element>
44 <xsd:element name="Mechanism" minOccurs="0">
45 <xsd:complexType>
46 <xsd:sequence>
47 <xsd:element name="id" type="xsd:string"/>
48 <xsd:element name="system" type="xsd:string"/>
49 </xsd:sequence>
50 </xsd:complexType>
51 </xsd:element>
52 </xsd:sequence>
53 </xsd:complexType>
54 </xsd:element>
55
56 <!--Uses-->
57 <xsd:element name="Uses">
58 <xsd:complexType>
59 <xsd:sequence>
60 <xsd:element name="Mechanism">
61 <xsd:complexType>
62 <xsd:sequence>
63 <xsd:element name="id" type="xsd:string"/>
64 <xsd:element name="system" type="xsd:string"/>
65 </xsd:sequence>
66 </xsd:complexType>
67 </xsd:element>
68 <xsd:element name="Configuration" maxOccurs="1">
69 <xsd:complexType>
70 <xsd:sequence>
71 <xsd:element name="id" type="xsd:string"/>
72 <xsd:element name="system" type="xsd:string"/>
73 </xsd:sequence>
74 </xsd:complexType>
75 </xsd:element>
76 </xsd:sequence>
77 </xsd:complexType>
78 </xsd:element>
```

```
79
80  <!--DerivesFrom-->
81  <xsd:element name="DerivesFrom">
82  <xsd:complexType>
83  <xsd:sequence>
84  <xsd:element name="Capability">
85  <xsd:complexType>
86  <xsd:sequence>
87  <xsd:element name="id" type="xsd:string"/>
88  </xsd:sequence>
89  </xsd:complexType>
90  </xsd:element>
91  <xsd:element name="Mechanism" minOccurs="0">
92  <xsd:complexType>
93  <xsd:sequence>
94  <xsd:element name="id" type="xsd:string"/>
95  <xsd:element name="system" type="xsd:string"/>
96  </xsd:sequence>
97  </xsd:complexType>
98  </xsd:element>
99  </xsd:sequence>
100 </xsd:complexType>
101 </xsd:element>
102
103 </xsd:choice>
104 </xsd:complexType>
105 </xsd:element>
106 </xsd:schema>
```

Listing 3.1: XML schema of causal graph data model.

We also crafted an example causal graph and its corresponding XML format that is based on our schema. This causal graph is shown as a graph in Figure 3.4 and in XML format in Listing 3.2. As the causality-based model is a novel approach, we could not find any documentation that captures the causal dependencies of trust notions, capabilities, mechanisms and configurations. Thus, this example approximates a design of data storage encryption using the Trusted Platform Module (TPM) version 2.0 [4]. It describes how the capability of disk encryption is derived from a software application named "DiskLocker". This software application is represented as a mechanism and it calls on various TPM 2.0 subsystems. These TPM 2.0 subsystems are represented as mechanisms. The "DiskLocker" software and some of the TPM 2.0 subsystem use configurations. For example, the configuration of "DiskLocker" states where the symmetric cryptographic key will be stored. In another example, the random number generator has to be configured with a seed value.

```
1  <causal_graph_data>
2
3  <ReliesOn>
```

```
<TrustNotion> <id>confidentiality</id> </TrustNotion>
<Mechanism> <id>cpe:/a:example:disklocker:1.0</id>
<system>PHD_MC355_004</system> </Mechanism>
</ReliesOn>

<DerivesFrom>
<Capability> <id>disk_encryption</id> </Capability>
<Mechanism> <id>cpe:/a:example:disklocker:1.0</id>
<system>PHD_MC355_004</system> </Mechanism>
</DerivesFrom>

<CallsOn>
<MainMechanism> <id>cpe:/a:example:disklocker:1.0</id>
<system>PHD_MC355_004</system> </MainMechanism>
<SubMechanism> <id>cpe:/a:tpm:subsystem_key_generation:2.0</id>
<system>PHD_MC355_004</system> </SubMechanism>
<SubMechanism> <id>cpe:/a:tpm:subsystem_nv_memory:2.0</id>
<system>PHD_MC355_004</system> </SubMechanism>
<SubMechanism> <id>cpe:/a:tpm:subsystem_rng:2.0</id>
<system>PHD_MC355_004</system> </SubMechanism>
<SubMechanism> <id>cpe:/a:tpm:subsystem_symmetric_engine:2.0</id>
<system>PHD_MC355_004</system> </SubMechanism>
</CallsOn>

<Uses>
<Mechanism> <id>cpe:/a:example:disklocker:1.0</id>
<system>PHD_MC355_004</system> </Mechanism>
<Configuration> <id>CCE-071015-1</id>
<system>PHD_MC355_004</system> </Configuration>
</Uses>
<Uses>
<Mechanism> <id>cpe:/a:tpm:subsystem_key_generation:2.0</id>
<system>PHD_MC355_004</system> </Mechanism>
<Configuration> <id>CCE-071015-2</id>
<system>PHD_MC355_004</system> </Configuration>
</Uses>
<Uses>
<Mechanism> <id>cpe:/a:tpm:subsystem_nv_memory:2.0</id>
<system>PHD_MC355_004</system> </Mechanism>
<Configuration> <id>CCE-071015-3</id>
<system>PHD_MC355_004</system> </Configuration>
</Uses>
<Uses>
<Mechanism> <id>cpe:/a:tpm:subsystem_rng:2.0</id>
<system>PHD_MC355_004</system> </Mechanism>
<Configuration> <id>CCE-071015-4</id>
<system>PHD_MC355_004</system> </Configuration>
```

```
51 </Uses>
52 <Uses>
53 <Mechanism> <id>cpe:/a:tpm:subsystem_symmetric_engine:2.0</id>
54 <system>PHD_MC355_004</system> </Mechanism>
55 <Configuration> <id>CCE−071015−5</id>
56 <system>PHD_MC355_004</system> </Configuration>
57 </Uses>
58 </causal_graph_data>
```

Listing 3.2: XML representation of the sample causal graph.

In this example, the ”DiskLocker” software uses symmetric cryptography to encrypt data. The symmetric cryptographic mechanism is provided by the TPM 2.0. ”DiskLocker” obtained a random number from the TPM 2.0 random number generator mechanism before the symmetric cryptographic mechanism of TPM 2.0 can be used. This random number is passed on to the TPM 2.0 key generation mechanism which produces a symmetric cryptographic key. This symmetric cryptographic key is then used with the symmetric cryptographic mechanism to encrypt data. To protect the symmetric cryptographic key, ”DiskLocker” stores it in TPM 2.0 using the TPM 2.0 non-volatile memory mechanism. The quality of the disk encryption capability is affected if ”DiskLocker” uses another random number generator that is not trusted. As a result, the trust notion of confidentiality that relies on the capability of disk encryption will be weaker than the situation where a trusted random number is used. Although the actual operation of such an implementation is more complex than that is described here, we decided to show just the necessary causal dependencies to ease practical application. Nevertheless, the causality-based model can be expanded to include more precise causal dependencies.

On the identity of mechanisms and configurations, a straightforward way to represent them is to refer to the Common Platform Enumeration (CPE) Reference[2] and the Common Configuration Enumeration (CCE) Reference[3]) of the National Institute of Standards and Technology. CPE is an organised naming scheme for computing systems, software, and packages. It is based on Uniform Resource Identifiers (URIs) and contains a formal name format that support name verification. CCE gives unique identifiers to system configurations to support correlation of configuration data. Since the example is an approximation, we assign dummy identifiers to support the simulation. We note that the TPM 2.0 is neither listed in CPE nor CCE. The TPM is typically implemented as one device and the mechanisms described in this example are subsystems of the TPM device. However, the TPM specification allows the exact implementation to vary. For example, only the SM4 symmetric cryptographic algorithm is allowed in certain geographical regions. Therefore, we identified the TPM subsystems as separate mechanisms and consequently they have their own configurations.

We worked with the Fedora 20 operating system running the Linux kernel version 3.19.8. For the MAP server, we reviewed the open source *irond* version 0.5.6 MAP server developed by the trusted computing research group at the Hochschule Hannover, University of Applied Sciences and Arts [4]. The latest version that sup-

[2] https://nvd.nist.gov/cpe.cfm
[3] https://nvd.nist.gov/cce/
[4] http://trust.f4.hs-hannover.de/

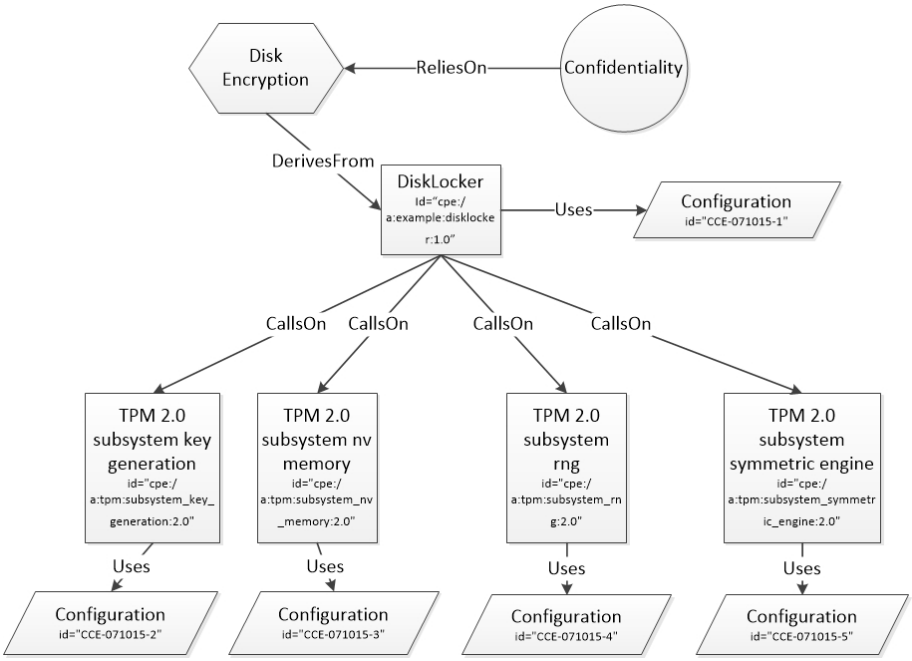


Figure 3.4: Pictorial representation of a sample causal graph.

ported IF-MAP version 2.2 specification was used. It was a Java program and refered to stored XML schemas. We added our schema to that of IF-MAP Metadata for Network Security. Then we examined the open source *ifmapj* version 2.3.0 Java library developed by the same team. We ran a MAP client program that made use of this Java library. This MAP client program encapsulates our example XML shown in Listing 3.2 and publishes it to the irond MAP server through an internal network. From this simulation, we managed to demonstrate how the causality-based model can be implemented on the TNC open architecture.

3.6 Trust Assessment

We developed an assessment rule to determine if a causal graph meets the requirement of a trust policy. A trust policy will consist of a set of assessment rules that lay out what are the identity and type of the vertices and edges required in a causal graph of a computing device before it can be trusted. We observed that the root of causal graphs will always be the vertices of Trust Notion. Thus, we can craft trust policies by starting with trust notions. To support this assessment, we formulated a rule that is based on determining the existence of a particular triplet in the causal graph. This is the most basic assessment rule and a trust policy will consist of a set of such basic assessment rules. This basic assessment rule is specified in the Backus Naur Form and it is shown in Listing 3.3 below.

```
1  (*U, RO, CO and DF are the short form for Uses, ReliesOn, CallsOn and
     DerivesFrom*)
2  (*ME, CF, TN and CP are the short form for Mechanism, Configuration,
     TrustNotion and Capability*)
```

```
3   <VertexType> ::= <ME> | <CF> | <CP> | <TN>
4   <SourceVertex>, <DestinationVertex> ::= <VertexType> <VertexID> | <
      VertexType> <id> <system>
5   <DependencyType> ::= <U> | <RO> | <CO> | <DF>
6   <Triplet> ::= "("<SourceVertex>"," <DependencyType>"," <
      DestinationVertex>")"
7   <BasicAssessmentRule> ::= "Is" <DestinationVertex> "∈ of a triplet
      containing ("<SourceVertex> "," <DependencyType>")?"
```

Listing 3.3: Specification of the basic assessment rule.

The specification begins by explaining the terms used. Line 3 states that the type of vertex can be a mechanism, a configuration, a capability or a trust notion. Line 4 then says that a target vertex is associated to an identity, and a system if the vertex type is a mechanism or a capability. The type of causal dependency is declared in line 5. Line 6 says that a triplet is projected as (SourceVertex, DependencyType, DestinationVertex). Hence, queries will be formulated around the concept of a triplet. Line 7 states a precise question on the existence of a destination vertex in a triplet. This question is the foundation of the assessment rule.

To develop a trust policy, we will have to understand how a mechanism interacts with another and what capability and trust notion does it enable. The trust policy will consist of numerous basic assessment rules that transverse through a causal graph and checking every branch from the root node to the leaf node. Figure 3.5 shows an example of a trust policy for the causal graph in Figure 3.4.

```
1 for $c in
2 /causal_graph_data
3 where
4 $c/CallsOn/MainMechanism/id="cpe:/a:example:disklocker:1.0"
5 return
6 if ($c/CallsOn/SubMechanism/id="cpe:/a:tpm:subsystem_key_generation:2.0")
      then
7 <result1> yes </result1>
8 else
9 <result1> no </result1>
```

Listing 3.4: XQuery command to check for a specific mechanisms that "DiskLocker" calls on.

The most direct way to assess a causal graph for trustworthiness is to process all the rules with a Boolean AND function. If the causal graph does not satisfy one rule, then the trust assessment will fail. However, we acknowledge that this assessment method is not flexible. If the trust assessment is to consider a range of values, then the rules can be assessed according to a Boolean Decision List [73]. However, we have to be careful with the complexity of the assessment process although we can deduce that the problem space is achievable in polynomial time if the number of rules is finite.

To implement the basic assessment rule, we investigated the use of the XQuery language. For example, if the query is "Is SubMechanism id="cpe:/ a:tpm:subsystem_key_generation:2.0" system="PHD_MC355_004" ∈ of a triplet

1. Is Capability id=”Disk Encryption” $\in$ of a triplet containing (TrustNotion=”Confidentiality”, RO)
2. Is Mechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004” $\in$ of a triplet containing (Capability id=”Disk Encryption”, DF)
3. Is Configuration id=”CCE-071015-1” system=”PHD_MC355_004” $\in$ of a triplet containing (Mechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004”,U)
4. Is SubMechanism id=”cpe:/a:tpm:subsystem_key_generation:2.0” system=”PHD_MC355_004” $\in$ of a triplet containing (MainMechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004”, CO)
5. Is Configuration id=”CCE-071015-2” system=”PHD_MC355_004” $\in$ of a triplet containing (Mechanism id=”cpe:/a:tpm:subsystem_key_generation:2.0” system=”PHD_MC355_004”, U)
6. Is SubMechanism id=”cpe:/a:tpm:subsystem_nv_memory:2.0” system=”PHD_MC355_004” $\in$ of a triplet containing (MainMechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004”, CO)
7. Is Configuration id=”CCE-071015-3” system=”PHD_MC355_004” $\in$ of a triplet containing (Mechanism id=”cpe:/a:tpm:subsystem_nv_memory:2.0” system=”PHD_MC355_004”, U)
8. Is SubMechanism id=”cpe:/a:tpm:subsystem_rng:2.0” system=”PHD_MC355_004” $\in$ of a triplet containing (MainMechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004”, CO)
9. Is Configuration id=”CCE-071015-4” system=”PHD_MC355_004” $\in$ of a triplet containing (Mechanism id=”cpe:/a:tpm:subsystem_rng:2.0” system=”PHD_MC355_004”, U)
10. Is SubMechanism id=”cpe:/a:tpm:subsystem_symmetric_engine:2.0” system=”PHD_MC355_004” $\in$ of a triplet containing (MainMechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004”, CO)
11. Is Configuration id=”CCE-071015-5” system=”PHD_MC355_004” $\in$ of a triplet containing (Mechanism id=”cpe:/a:tpm:subsystem_symmetric_engine:2.0” system=”PHD_MC355_004”, U)

Figure 3.5: Example of a trust policy.

containing (MainMechanism id=”cpe:/a:example:disklocker:1.0” system=”PHD_MC355_004”, CO)”, we can issue the XQuery command in Listing 3.4. The return result will be ”yes” in this example.

3.7 Related Works

As far as we know, our work on the causality-based model is the first attempt at describing the capabilities of a computing device. Although this causality-based model can appear similar to the ontology of Chapter 5, there are key differences because they are intended for different purposes. First, the causality-based model intends to describe the trustworthiness of a computing device. Hence, its description covers the causal dependencies between trust notions, capabilities, mechanisms and configuration. This description pertains to a specific implementation and its configurations. On the other hand, the ontology of Chapter 5 is a mapping of a class of computing device. In that chapter, the ontology maps the class of computing device secured with TPM 2.0 and does not include the applied configurations. Thus, that ontology serves as a standard vocabulary for experts to share information on TPM 2.0 with developers of trusted systems. Secondly, the structure of the causality-based model depends on the unambiguous definitions of causal dependencies and has rules governing its use while the structure of an ontology is founded on class hierarchies and user defined semantic relations. As a result, the ontology is not as robust as the causality-based description when used to describe the trustworthiness of a computing device. However, these two methods can be complementary. For example, the initial development of a causality-based description of a specific implementation can refer to the ontology of the class that this computing device belongs to.

3.8 Summary

We revisit the requirements mentioned in Section 3.2. On the requirement to describe the capabilities of a computing device that give rise to its trustworthiness, we have developed a causality-based model that describes the causal dependencies between trust notions, capabilities, mechanisms and configurations. To make the model clear and easy to understand, we have given the basic definitions of the model and used set theory to further clarify the definitions. We then transformed the causality-based model into a causal graph for the purpose of data representation. Additional definitions are given for the graph model to show clearly how it can be used. During implementation, we have gained insights into how the causality-based model can be extended to the schemas used by the MAP server of the TNC open architecture and gave an example of a causal graph and its corresponding XML format. We also explained how to carry out trust assessment of the XML based causal graph. These practical exercises show that the requirement to support digital representation of this model is met. Most importantly, this causality-based model enables the descriptive property of Grawrock’s trust language. The trust notions, capabilities, mechanisms, configurations and their causal dependencies defined in Section 3.3 are the grammar of the trust language. On the

other hand, the graph definition from Section 3.4 offers a standard form to ensure that the structure of the trust language is consistent.

Chapter 4

Provenance-based Attestation for Trustworthy Computing

4.1 Introduction

Attestation, in the context of trustworthy computing, is carried out with the intention of uncovering if a computer can behave in an expected manner when given a specific task to perform. Thus, the evidence given for trust assessment often identifies the software stack running on the attesting computing and if they are in a state that can be expected to perform a specific task in a trusted manner. This notion of expected behavior is viewed by the Trusted Computing Group (TCG) [49] as the tenet of trustworthiness.

This chapter follows on from the previous chapter introducing our causality-based model. When a causality-based description of the trustworthiness of a computing device is given, the first logical step is to evaluate if the description meets certain requirements. If the evaluation is successful, the next logical step is to check if the mechanisms and configurations stated in the causality-based description behave in an expected manner to give rise to the capabilities and trust notions. This step entails the attestation of the identity and state of these mechanisms and configurations.

The prevailing practice of attestation is to compare the integrity measurements of binaries that make up the software stack of a computer against pre-defined trusted values. This technique is known as binary attestation [5]. When the computer is booting up, integrity measurements are obtained by passing each binary of the software stack through a hash function. The resulting hash digest is unique to this particular software binary and its state. In many implementations, the integrity measurements are stored into a security device known as the Trusted Platform Module (TPM) which is specified by the TCG [92].

Besides measuring the key components of the software stack, integrity measurement can be extended to cover important configuration files such as the password file. If all the integrity measurements of the key software binaries and configurations match pre-defined trusted values, then it can be said that the computer is trustworthy. When a software binary is altered and the pre-defined trusted value is not updated under authorization to reflect the change, then the resulting hash digest will be different from the pre-defined value and, thus, the trustworthiness of the computer will be cast in doubt.

However, the implementation of binary attestation has faced many challenges [74]. The main problem is that the integrity measurement value contains little information. This integrity measurement value is only made meaningful when there is a

reference value which reflects the identity and state of the source component. If no such reference value is available, then binary attestation fails as there are no other means to check the trustworthiness of the component. The situation of no reference value can arise when a component in the measured software stack has changed but the change is not reflected in the reference database. Although the change could be malicious, it could also be the case that the change is legitimate and the component remains trustworthy.

There have been efforts to develop alternatives to binary attestation. Property-based attestation was proposed by Sadeghi and Stuble [74]. The main concept is that attestation should verify if a computer possesses properties to fulfill certain requirements of the party who asks for attestation. Its implementation relies on a trusted third party that certifies the properties on the attesting computer. However, this approach is challenged by the broad definition of "property". On the other hand, Halder et al. [29] proposed semantic remote attestation which leverages program analysis to determine whether a program's execution will satisfy certain properties. Similar to property-based attestation, this approach is thwarted by the definition of "property". Meanwhile, Alam et al. [1] presented a model-based behavioral attestation where the trustworthiness of a target computer is based on the behaviour of its policy model. But the semantics of this proposal are high level and they require transformation to a low level policy that is implementable. Thus, finding workable alternatives to binary attestation is still a very much open research problem.

In view of these challenges, we investigate and develop a design for using provenance data as evidence of trustworthiness in the attestation of a computer. This chapter reports on this work. In Section 4.2, we articulate how provenance data can be used as evidence of trustworthiness. Section 4.3 describes the goals that guide the development of this design. Section 4.4 gives an overview of this provenance-based attestation design. This is followed by Section 4.5 which explains the collection and representation of provenance data. The specification of trust evaluation rule and the evaluation algorithm are covered in Section 4.6. We share our experience at building the key mechanisms of this design in Section 4.7. Threat modelling of this design is given in Section 4.8 and a mitigation to the identified threats is given in Section 4.9. Related works are discussed in Section 4.10 and the summary of this chapter is in Section 4.11.

4.2 Provenance Data as Trust Evidence

Provenance is the documentation of the origin of an object and the processes that cause it to be in this current state [53]. Recording and storing the provenance data of an object enable a party to evaluate the contextual and circumstantial evidence of the origin of the object and its transformation to the current form. The semantics of provenance describe how the state of an object is caused by an effect. In other words, the semantics of provenance captures the causal dependencies between the different states of an object and other elements. This description for dependency path is often written as a triplet (x,y,z) where component x was related to another component z through a property y. It can also be expressed as a directed acyclic graph where the nodes are x and z while the directed edge is y. Hence, we can

conjecture that the provenance of an object is composed of multiple provenance data and they form a larger directed acyclic graph with numerous nodes that are connected by directed edges.

In scientific research, provenance has been traditionally associated with e-Science that makes use of provenance systems to track data objects used and generated in computer simulated experiments [47]. Another popular implementation is the use of provenance data models to track the origin, use and transformation of objects and resources on the Web [54]. However, we argue that the principle behind provenance is applicable to the attestation of a computer. In this context, software or configuration files in a computer are treated as an object. In the same vein, the provenance data of this object will contain information about its origin and how it is changed over time. If it can be ascertained that the object's origin is reputable, the change over time is authorized by a legitimate party and the change parameters are reliable, then there is a high degree of confidence in the identity of the object and the expectation that it can perform the task intended by its creator. This outcome is compatible with the trust notion of expected behavior defined by the TCG. Thus, the provenance data of key components of a computer can be used as evidence during attestation to back up its claim of trustworthiness.

The most important advantage that provenance data has over an integrity measurement value is that it contains more information. When an integrity measurement value does not match a reference value, it is difficult to trace what causes the software to change to an unknown state. However, it is relatively straightforward in the case of a provenance data as the semantics would have connected the information together. This property of the provenance data also makes it better than using a system log for identifying the cause, whether malicious or legitimate, as a system log records events individually and does not connect them together. We will describe this advantage further in Section 4.6.

The following is an example of an application of provenance-based attestation. In an enterprise that has a Bring-Your-Own-Device (BYOD) policy [3], computers are not always connected to the corporate computer network and any changes to these computers can happen without the knowledge and authorization of the computer network administrators. Thus, if binary attestation is its primary form of attestation, provenance-based attestation can augment the trust evaluation when a reference integrity measurement value is not available or when an integrity measurement value does not match the reference value.

4.3 Design Goals

The following two goals are proposed to guide the development of the provenance-based attestation design.

Design Goal 1 : The provenance data of a component has to contain suitable information such that it can be used by another party to make a trust evaluation. This requires a data model that could describe the provenance data of a computer component in a manner so that this provenance data could be used as evidence of trustworthiness for the attestation of a computer. In addition, this data model is required to represent the provenance data of a computer component in a structured format so that this format can be read by another party. This design goal also alludes to a

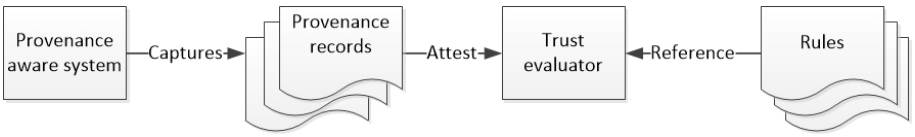


Figure 4.1: A design of provenance-based attestation.

system design that describes how provenance data is collected and used by a trust evaluator.

Design Goal 2 : Rules must be able to be created to guide the trust evaluation of a provenance record. To evaluate provenance data for evidence of trustworthiness, the evaluator should assess the presented data according to a set of pre-defined rules. Hence, the rules are to be developed with reference to the kind of provenance data that could be represented. For a thorough evaluation, the rules have to examine all possible causal dependencies between the component and other agents and activities of the host computer and the related attributes. Hence, the evaluation will have to process the various rules against the provenance data and produce a result to indicate the trustworthiness.

4.4 A Design for Provenance-based Attestation

Our design in Figure 4.1 describes the key modules in provenance-based attestation. We assume that the attesting computer can reach the trust evaluator over a computer network.

The following paragraphs give a general description of the modules in this design. Detailed descriptions and designs are presented in Sections 4.5 and 4.6.

Provenance Aware System: A provenance aware system is a computer system that contains automated mechanisms for capturing provenance data related to a target object. A target object could be a software binary or a configuration file. It is crucial that these mechanisms are deployed at strategic points in the computer system so that they are triggered whenever a change is made to a target object. An example of such mechanism would be software code inserted into an installer such as *rpm* [1] and *yum* [2]. Another example where these mechanisms can be inserted is text editors such as *vim* [3] and *gedit* [4].

Besides implanting such mechanisms at the application level of the software stack, system calls occurring at kernel level could also be monitored. For example, when a system call such as *vfs.write* is activated, a mechanism can be activated to capture provenance data related to this event. A provenance aware system can be configured to monitor the same components measured by binary attestation. Thus, when binary attestation fails, the related provenance data can be provided to augment the trust evaluation process.

Provenance Record: When a target object is changed in a provenance aware system, provenance data related to this instance of change will be captured as a prove-

[1] http://www.rpm.org/
[2] http://yum.baseurl.org/
[3] http://www.vim.org/
[4] https://wiki.gnome.org/Apps/Gedit

nance record. The provenance record presents the captured provenance data in a structured format so that it can be evaluated by another party to determine if the target object is trustworthy. In additional, a provenance record is updated in real time and hence it always reflects the up to date status of the target object. As numerous target objects can be monitored in a provenance aware system, there can be several provenance records existing in the attesting computer. Every provenance record is identified by tagging it to the target object. During attestation, the party conducting the trust evaluation can request the attesting computer to provide provenance records of interest. In this design, the trustworthiness of a component depends on the evaluation of its provenance record. Therefore, provenance records should be protected from tampering. In addition, information in the provenance records could reveal the composition of a computer. Hence, they should not be read by an unauthorized entity.

Trust Evaluator: After the attesting computer sends over the requested provenance record, the trust evaluator will examine the record for evidence that indicates the trustworthiness of the target object. The evaluation takes in the provenance record and a set of rules as inputs, examines the provenance record with reference to the rules, and outputs a result about the trustworthiness of the target object.

Rules: The rules are crafted with the intention of checking a provenance record for certain provenance data that indicate the trustworthiness of the target object. The rules can check on the dependencies between the target object and other elements of the attesting computer. The rules can also check on the attributes of these elements. As there can be numerous provenance records tagged to different target objects, there will be different sets of rules that are meant to evaluate provenance records of different target objects. If no rule set is available to evaluate a particular provenance record, then the trust evaluation can prompt the administrator for intervention.

4.5 Collecting and Representing Provenance Records

To identify what provenance data to collect, we analyze what kind of provenance data convey the TCG's central notion of expected behavior. A provenance record of a target object contains provenance data related to an instance of change. The record will later allow an evaluator to ask questions on provenance as part of the trust evaluation process. Some examples of questions are: "What was the activity that produced this version of software?", "Who authorized this software change?" and "Did the activity that produced this version of software use the correct inputs?" Answering these questions requires analyzing the provenance record of the target object. Hence, a provenance aware system has to capture, in a provenance record, relevant information that could be used to answer these questions about the change in state of the target object.

There are several data models for representing provenance data. The Open Provenance Model (OPM) [53] is a well-known provenance data model published by the World Wide Web Consortium (W3C) while the Dublin Core model [97] is a generic metadata standard that can be customised to represent provenance data. In April of 2013, the provenance working group of W3C published the PROV data model [54]. This newer data model supersedes the Open Provenance Model and

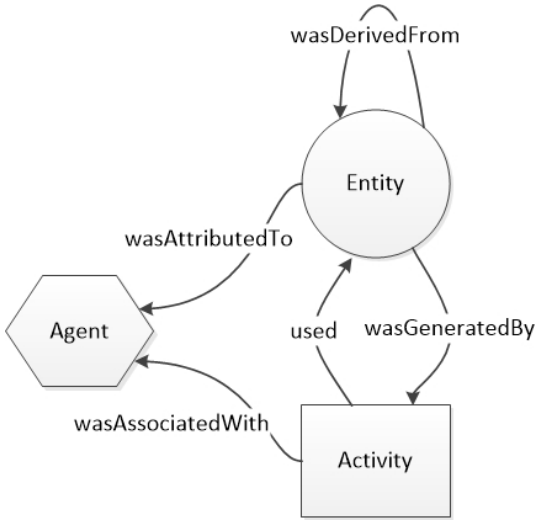


Figure 4.2: A graph describing the PROV data model redrawn from [54].

can be mapped to the Dublin Core model. We reviewed the specifications of the PROV data model and found that the semantics are useful for answering the provenance questions discussed in the previous paragraph. No change or extension to the PROV data model is required. Although the Dublin Core model can be adapted to represent provenance data, we decided to take a more direct approach and use the PROV data model in this work. Nevertheless, we do not foresee any difficulties with using the Dublin Core model in the design described in Section 4.4.

The PROV data model defines the representations of the entities, agents, activities and their relationship in the production of an object. We use the graph in Figure 4.2 to give an overview of the PROV data model. Brief explanations of the nodes and edges in the PROV provenance graph in the context of provenance-based attestation are given in the following paragraphs. This is to aid the understanding of the designs described in this chapter. A more detailed explanation of PROV data model can be obtained from [54].

Entity: An entity is an object and examples are a software binary and a configuration file.

Activity: In the context of provenance-based attestation, activities refer to the process that changes an entity. For example, the rpm installer deploys a new software version and removes the outdated version from a computer system.

Used and wasGeneratedBy: An activity can either use an entity or generate an entity. For example, coding a program brings a program into existence. Another example is patching an older software version to a newer software version. Note that these two edges are directed and only exist between an entity node and activity node. Past tense is used for these two terms because provenance is about events in the past.

Agent, wasAssociatedWith and wasAttributedTo: An agent has a relationship with an activity in such a way that the agent has responsibility for the activity taking place. This relationship of responsibility is termed wasAssociatedWith. An agent can be a decision making software, a person or an organization. On the other hand,

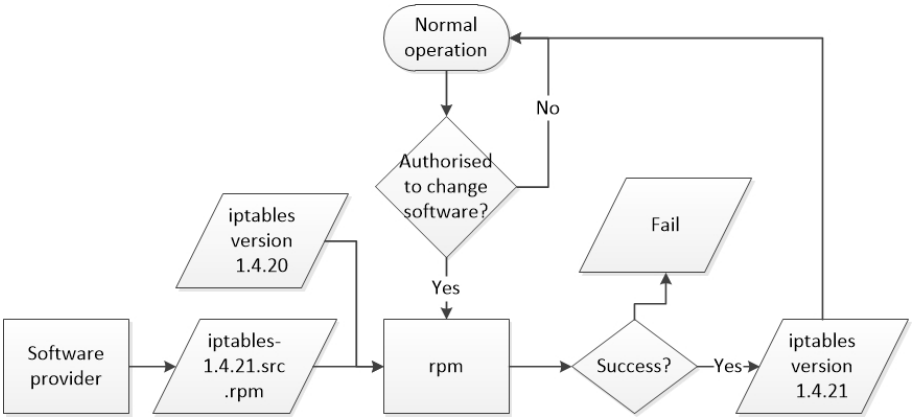


Figure 4.3: Flowchart for upgrading iptables using rpm.

when an agent has responsibility for an entity, this relationship is termed wasAttributedTo and it means that the agent is responsible for an activity that generated that entity. wasAttributedTo can also be used as an abstract representation to represent the dependency that an entity was generated by an activity and this activity was associated with an agent. Note that these two edges are directed and wasAssociatedWith is used with an agent and an entity while wasAttributedTo is used with an agent and an activity.

wasDerivedFrom: An entity can be derived from another entity. For example, this version of configuration is derived from an earlier version. wasDerivedFrom can be used as an abstract representation of the event when an activity uses an entity to generate another entity.

A node or an edge in the PROV provenance graph can have attributes to describe certain contextual information. These attributes can possess a value. For instance, an authority agent has an identity of "computer administrator" and it has a "type" attribute which has the value "person". Meanwhile, an entity is always drawn as a circle, an activity is a box and agents are hexagon shaped. In the PROV data model, one can express a dependency path as a triplet, for example, (Entity, wasGeneratedBy, Activity).

We will use the example of upgrading the *iptables* [5] program using the *rpm* package manager to show how a provenance record for this software can be represented. *iptables* is commonly found in the Linux based operating system and it controls the kind of network connectivity the operating system can have. Figure 4.3 shows a generalized work flow for the process of using *rpm* to upgrade the *iptables* program from version 1.4.20 to version 1.4.21. The flowchart shows that the source file is obtained from a software provider and the upgrading is approved by an authority. The *rpm* manager uses the source file iptables-1.4.21.src.rpm.

In provenance-based attestation, we wish to capture provenance data that can be used to assist with trust evaluation. In this example, we are interested in the causal dependencies amongst *iptables, rpm*, authority and software provider. We are also interested in their attributes. Hence, the provenance record has to contain this information. The provenance graph for this example is given in Figure 4.4.

[5] http://www.netfilter.org/projects/iptables/

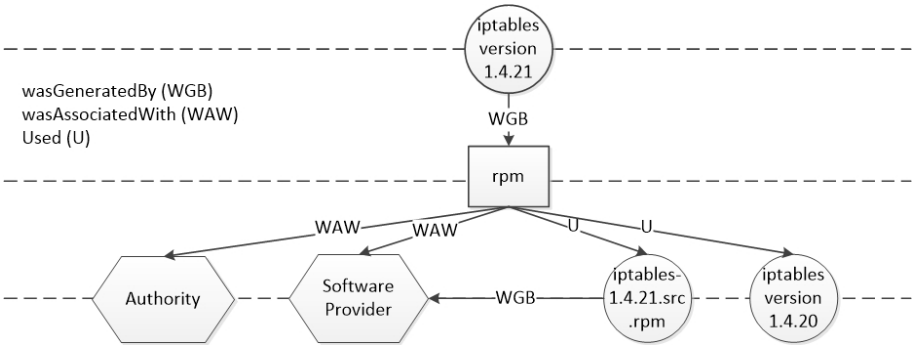


Figure 4.4: Provenance graph of upgrading iptables.

The rpm package manager is defined as an activity. The entities consist: of the source file iptables-1.4.21.src.rpm; the software to be upgraded, iptables version 1.4.20 and the software after upgrading; iptables version 1.4.21. The authority agent has the responsibility of approving the upgrading while the software provider agent has the responsibility of delivering the source file.

While the provenance graph provides a visual description of the provenance record, the provenance aware system actually produces a textual provenance record. The PROV data model has provision for using several textual schemas such as Terse RDF Triple Language (Turtle) [6] and eXtensible Markup Language (XML) to represent a provenance record in text format. We found that both Turtle and XML have similar capabilities but we chose to use XML schema as it is more widely used and supported by many application tools. Listing 4.1 shows the XML based provenance record for upgrading *iptables*. The XML based provenance record includes more information, such as time, role and attributes, which are not shown in the provenance graph.

The XML based provenance record for this example of upgrading *iptables* begins by tagging it to the target object and stating the namespace used (lines 1 to 4). It then proceeds with the listing of nodes by their type: entity, agent or activity (lines 5 to 25). Each node has an identity name. For example, line 6 describes the ”iptables version 1.4.21” entity. This entity does not have any attributes. Referring to lines 10 to 12, the ”rpm” activity has a value which is the code signing certificate. This code signing certificate can be verified by the trust evaluator later to check on the authenticity of the *rpm* installer. Meanwhile, from lines 13 to 25, the ”authority” agent has attributes that describe its name, email address and an identity number while the ”software provider” agent has attributes that describe its name and URL web link.

The relationships between the nodes are next described (lines 26 to 52). These can be considered as triplets that describe the causal dependencies. For example, lines 27 to 30 simply describe that ”rpm” used ”iptables-1.4.21.src.rpm”. In addition, lines 40 to 46 describe that ”rpm” was associated with ”authority”. The role of this association is to approve *iptables* patch to 1.4.21. There is a value which is the approval identity number and the time of approval is recorded as well. The provenance record ends with the additional information of the start and end time of the patching process (lines 54 to 61). This information can be utilized if the trust

evaluator wishes to find out when this event took place.

```
1  <?xml version='1.0' encoding='UTF-8'?>
2  <prov: iptables version 1.4.21 provenance record
3  xmlns:prov=http://www.w3.org/ns/prov#
4  xmlns:foaf="http://xmlns.com/foaf/0.1/">
5  <!--Entities-->
6  <prov:entity prov:id="iptables version 1.4.21"/>
7  <prov:entity prov:id="iptables version 1.4.20"/>
8  <prov:entity prov:id="iptables-1.4.21.src.rpm"/>
9  <!--Activities-->
10 <prov:activity prov:id="rpm">
11 <prov:value> Code Signing Certificate 03
       ce7ecd0cc462b0b0bef08d400f5a39 </prov:value>
12 </prov:activity>
13 <!--Agents-->
14 <prov:agent prov:id="authority">
15 <prov:type> prov:Person </prov:type>
16 <foaf:givenName> Admin </foaf:givenName>
17 <foaf:mbox> mailto:admin@example.org </foaf:mbox>
18 <prov:value> ID_Admin_4567 </prov:value>
19 </prov:agent>
20 <prov:agent prov:id="software provider">
21 <prov:type> prov:Organisation </prov:type>
22 <foaf:givenName> Netfilter </foaf:givenName>
23 <foaf:homepage> http://www.netfilter.org/projects/iptables/
24 downloads.html </foaf:homepage>
25 </prov:agent>
26 <!--Relations-->
27 <prov:used>
28 <prov:activity prov:ref="rpm"/>
29 <prov:entity prov:ref="iptables-1.4.21.src.rpm"/>
30 </prov:used>
31 <prov:used>
32 <prov:activity prov:ref="rpm"/>
33 <prov:entity prov:ref="iptables version 1.4.20"/>
34 </prov:used>
35 <prov:wasGeneratedBy>
36 <prov:entity prov:ref="iptables version 1.4.21"/>
37 <prov:activity prov:ref="rpm"/>
38 </prov:wasGeneratedBy>
39 <!--Responsibilities and Attributions-->
40 <prov:wasAssociatedWith>
41 <prov:activity prov:ref="rpm"/>
42 <prov:agent prov:ref="authority"/>
43 <prov:role>approve iptables patch to 1.4.21</prov:role>
44 <prov:value>Approval ID Z7890</prov:value>
45 <prov:time>2014-09-30T14:34:00</prov:time>
```

```
46 </prov:wasAssociatedWith>
47 <prov:wasAssociatedWith>
48 <prov:activity prov:ref="rpm"/>
49 <prov:agent prov:ref="software provider"/>
50 <prov:role>provide iptables-1.4.21.src.rpm </prov:role>
51 <prov:time>2014-09-30T14:33:00</prov:time>
52 </prov:wasAssociatedWith>
53 <!--Time-->
54 <prov:wasStartedBy>
55 <prov:activity prov:ref="rpm"/>
56 <prov:time>2014-09-30T14:35:00</prov:time>
57 </prov:wasStartedBy>
58 <prov:wasEndedBy>
59 <prov:activity prov:ref="rpm"/>
60 <prov:time>2014-09-30T14:36:00</prov:time>
61 </prov:wasEndedBy>
62 </prov: iptables version 1.4.21 provenance record >
```

Listing 4.1: XML based provenance record for patching iptables.

The main benefit of this XML based provenance record over commonly found records of change events in computer systems such as syslog is that the required provenance data are put together in a structured format and connected together meaningfully . Although it can be argued that the required provenance data can be reconstructed by trawling through logs of a computer system, there is a risk that certain provenance data is not captured at that moment in time and this results in an incomplete provenance record.

4.6 Trust Assessment of Provenance Records

A rule set is required as a reference during the trust evaluation of a provenance record. This rule set is a manifestation of the questions an evaluator will ask to ascertain the trustworthiness of the target object. There are two categories of questions: one that asks about the causal dependencies between the target object and other elements of the attesting computer and the other that asks about the attributes of the related elements. Consequently, there is a rule type for each category of questions. With this requirement in mind, we formulate a rule specification grammar to guide the expression of the rules in pseudo code. Listing 4.2 shows the rule specification grammar expressed in Backus Naur Form.

This rule specification grammar begins by explaining the terms used. Line 6 states that <NodeType> can either be an entity, activity or agent. Line 7 explains that both <QueryNode> and <TargetNode> are composed of the type and identity of node. <AttributeNode> at line 8 has the additional fields that describe the attribute identity and content. Line 9 defines that <DependencyType> can be used, wasGenerateBy, wasDerivedFrom, wasAttributedTo or wasAssociatedWith. <Dependency> at line 10 refers to the type of dependency and its attributes. Hence, it has the additional fields of attribute identity and content.

There are two rule types. The causal dependency type rule specified in line 11 is developed around the concept of expressing a causal dependency path as a triplet which is explained in Section 4.2. Essentially, the rule asks if a destination <QueryNode> can be found in the causal dependency path where <TargetNode> is the source and <Dependency> is the edge. An example is : Is ACT rpm $\in$ (ENT iptables version 1.4.21 , WGB) ? This example checks if the triplet (iptables version 1.4.21, wasGeneratedBy, rpm) is present in the provenance record. The other type of rule is the attribute rule described in line 12. The attribute rule asks if a node contains an attribute of certain identity and if the content of the attribute has a particular value. An example of an attribute rule is : Is value $\in$ AGT Authority $\cap$ value = ID_Admin_4567 ?

```
1  (*U, WGB, WDF, WAT and WAW are the short form for used,
2  wasGeneratedBy, wasDerivedFrom, wasAttributedTo and
3  wasAssociatedWith*)
4  (*ENT, ACT and AGT are the short form for Entity, Activity
5  and Agent*)
6  <NodeType> ::= <ENT> | <ACT> | <AGT>
7  <QueryNode>, <TargetNode> ::= <NodeType> <NodeID>
8  <AttributeNode> ::= <NodeType> <NodeID> <AttributeID> <
       AttributeContent>
9  <DependencyType> ::= <U> | <WGB> | <WDF> | <WAT> | <WAW>
10 <Dependency> ::= <DependencyType> <AttributeID> <AttributeContent>
11 <DependencyRule> ::= "Is" <QueryNode> "∈ ("<TargetNode> "," <
       Dependency> ")?"
12 <AttributeRule> ::= "Is" <AttributeID> "∈" <AttributeNode> "∩" <
       AttributeContent> "=" <value> "?"
```

Listing 4.2: Rule specification grammar for trustworthy evaluation of provenance record.

To develop specific dependency rules and attribute rules, we will have to understand the work flow that describes the change to a target object. An example of work flow is given in Figure 4.3. Thereafter, we refer to the work flow and develop a provenance graph of which an example is given in Figure 4.4. We can visualize Figure 4.4 as a directed rooted tree and the leaf nodes always have a parent node that describes the activity which produces the root node. The root node will refer to the target object monitored by the provenance aware system. This is an important observation. When we analyze other provenance records such as those about a change to a configuration file, we could make similar observations as well.

From Listing 4.1, we can see that the provenance record contains attribute data for the following nodes: rpm, Authority, Software Provider. Thus, by inspecting Figure 4.4 and Listing 4.1 simultaneously, we can develop a set of specific dependency rules and attributes rules to evaluate this particular provenance record and the development of rules starts with the examination of the root node before ending with the leaf nodes. Listing 4.3 shows the rule set for evaluating the provenance record given in Listing 4.1. The rule set starts with a dependency rule covering the root node and then through the branch node and eventually the leaf nodes. As stated in the previous paragraph about similar observations of the directed rooted

graph in the provenance record of both software binary and configuration file, the structure of the rule set in Listing 4.3 can be the template for their evaluation.

1 Is ACT rpm $\in$ (ENT iptables version 1.4.21 , WGB) ?
2 Is AGT Authority $\in$ (ACT rpm , WAW) ?
3 Is AGT Software Provider $\in$ (ACT rpm , WAW) ?
4 Is ENT iptables 1.4.21.src.rpm $\in$ (ACT rpm , U) ?
5 Is ENT iptables version 1.4.20 $\in$ (ACT rpm , U) ?
6 Is AGT Software Provider $\in$ (ENT iptables 1.4.21.src.rpm , WGB) ?
7 Is value $\in$ ACT rpm $\cap$ value = Code Signing Certificate 03 ce7ecd0cc462b0b0bef08d400f5a39 ?
8 Is value $\in$ AGT Authority $\cap$ value = ID_Admin_4567 ?
9 Is homepage $\in$ AGT Software Provider $\cap$ homepage = http://www.netfilter.org/projects/iptables/downloads.html ?

Listing 4.3: A set of dependency and attribute rules.

The most direct method to evaluate the provenance record given in Listing 4.1 and using the rule set from Listing 4.3 is to simply process all the rules using the Boolean AND function and the function starts with the rules related to the root node and ends with rules related to the leaf nodes. If the provenance record does not satisfy any of the rules, then the function will output that the provenance record does not attest to the trustworthiness of the target object iptables version 1.4.21. However, we acknowledge that this evaluation method is rather straightforward and could face issues in practical implementation as there could be multiple computer configurations. Therefore, as an example, if we are flexible with the installer software and trust yum as well, then the Boolean function for this scenario will be in disjunctive normal form (DNF) where the outcome of the evaluation depends on either the rules for rpm or the rules for yum.

If the requirement is for the trust evaluation to consider a range of values, for example, trusting a range of software installers instead of just rpm, then the trust evaluator can use the Boolean Decision List [73] to address this challenge. Meanwhile, in another example of the flexibility offered by this trust evaluation approach, the trust evaluator can only insist that rules 2, 3, 7, 8 and 9 are met. In this case, the trust evaluator is satisfied as long as the change to the target object is authorized and obtained from an approved source. Moreover, this approach can be used together with techniques developed for eXtensible Access Control Markup Language (XACML) [60] to provide for policy based trust evaluation. Therefore, we can conclude that these two rule types are the ”basic units” of a rule set used in the evaluation of a provenance record and they can be employed together with various techniques to allow for more flexible evaluation of a provenance record. However, the trust evaluator has to be careful with the complexity of the evaluation process although we can deduce that the problem space is achievable in polynomial time if the number of rules is finite. In addition, this approach to trust evaluation in provenance based attestation has an advantage over binary attestation. From Section 4.1, we understand that, in binary attestation, if the integrity measurement value does not match the reference value, the evaluation cannot proceed further. However, we show in this section that the trust evaluation in provenance-based attestation is versatile and can be adapted to deal with different situations.

```
[root@localhost iol]# cat wgb.stp
#!/usr/bin/stap

probe vfs.write
{
  # dev and ino are defined by vfs.write
  if (dev == MKDEV($1,$2) # major/minor device
      && ino == $3)
    printf ("<prov:wasGeneratedBy> <prov:entity prov:ref=0x%x/%u> <prov:activity prov:ref=%s(%d)> </prov:wasGeneratedBy>\n",
      dev, ino, execname(), pid() )
}
[root@localhost iol]#
[root@localhost iol]#
[root@localhost iol]#
[root@localhost iol]# stat -c '%D %i' /etc/test.txt
fd01 691452
[root@localhost iol]#
[root@localhost iol]#
[root@localhost iol]# stap wgb.stp 0xfd 0x01 674171 /etc/test1.txt
<prov:wasGeneratedBy> <prov:entity prov:ref=0xfd00001/674171> <prov:activity prov:ref=vi(3296)> </prov:wasGeneratedBy>
_
```

Figure 4.5: A screen shot of System Tap running a monitoring script.

4.7 Proof of Concept

To build a provenance aware computer system, we explored two approaches. In the first approach, we instrumented all software applications that are capable of altering the trustworthy state. In the second approach, we inserted program scripts at the kernel space to monitor actions that could alter the trustworthy state. We worked with the Fedora 20 operating system running the Linux kernel version 3.17.4. On the instrumentation of a software application, we reviewed the source code of the widely used text editor vim and inserted scripts using the autocmd function. This method allowed us to capture provenance data related to the relationships used, wasGeneratedBy and wasDerivedFrom. For instrumentation at the kernel space, we explored the tool System Tap version 2.6 [6] and developed a script to monitor the system call vfs.write. When this system call was activated, the monitoring script will automatically capture and record the provenance data related to used, wasGeneratedBy, wasAttributedTo and wasAssociatedWith. An example of using System Tap is given in Figure 4.5. In this example, the script "wgb.stp" was ran with System Tap to monitor any vfs.write to the file "/etc/test.txt". When the program *vi* carries out a vfs.write on this file, the provenance data of this event will be captured.

Thus, we show that provenance information can be captured and recorded for use later to support the trust evaluation of an object. However, we found that both approaches are not exclusive and should be employed together to produce the most complete provenance record. We also observed that the ability of a provenance aware system to capture all provenance data of a target object depends on how comprehensively the system is instrumented. In spite of this, we were unable to monitor objects residing at the BIOS level as the instrumentations occur at the OS level.

Meanwhile, we wanted to find out how provenance-based attestation will be accepted by professionals. First, we presented the provenance-based attestation design to a group of 10 people who have worked in information security for at least a year and introduced to them the rule specification grammar. Then a quick survey was conducted at the end of the 30 minute session and 8 of them stated that they

[6] https://sourceware.org/systemtap/

could understand the rule specification grammar and found its usage during trust evaluation to be easy. Although the sample size is small, this is a good indication that the majority of information security professionals will not face a steep learning curve and will be able to develop rules to evaluate provenance records.

To build the trust evaluator, we investigated the use of the XQuery language to evaluate the XML-based provenance record for evidence that support the trustworthiness of its tagged object. We earlier summarised that there are two types of rules for the evaluation of a provenance record. The dependency rule checks the existence of a certain causal dependency path. For example, the object must be changed by an authorized software. On the other hand, the attribute rule checks the presence of certain attributes. For example, the identity is of a particular value. These requirements are collectively expressed in a set of rules for a target object. We then transformed the rules into the XQuery language and ran it against a sample provenance record. Listing 4.4 shows the corresponding XQuery for the dependency rule "Is ACT rpm $\in$ (ENT iptables version 1.4.21 , WGB) ?" and Listing 4.5 shows the corresponding XQuery for the attribute rule "Is value $\in$ Agent Authority $\cap$ value = ID_Admin_4567 ?".

```
1 for $ pr in
2 doc("prov_rec.xml")/prov:iptables_version_1.4.21_provenance_record
3 where
4 $pr/prov:wasGeneratedBy/prov:entity/@prov:ref="iptables_version_1.4.21"
5 return
6 if ($pr/prov:wasGeneratedBy/prov:activity/@prov:ref="rpm") then
7 <result1>success</result1>
8 else
9 <result1>fail</result1>
```

Listing 4.4: Dependency rule expressed as XQuery.

```
1 for $pr in
2 doc("prov_rec.xml")/prov:iptables_version_1.4.21_provenance_record
3 where $pr/prov:agent/@prov:id="authority"
4 return
5 if ($pr/prov:agent/prov:value="ID_Admin_4567") then
6 <result2>success</result2>
7 else
8 <result2>fail</result2>
```

Listing 4.5: Attribute rule expressed as XQuery.

The XQueries contain a for loop that searches the xml document for a specific markup. In the case of the dependency rule, it searches for the markup containing wasGeneratedBy, entity and iptables version 1.4.21 (line 4). Similarly for the attribute rule, it searches for the markup containing agent and authority. If the required markup is not found, the query will return a message stating that it is an empty sequence. When the required markup is found, the XQuery algorithm extracts the content and compares it to the value specified in the rule (line 6 of Listing 4.4 and line 5 of Listing 4.5). If the content matches the value specified in the rule, then the XQuery algorithm will return success. Else, the XQuery algorithm

will return fail. The interrogation is able to identify either the presence or absence of dependency path or attribute as specified in the rule. Hence, we prove that a provenance record can be evaluated against a pre-defined rule for the purpose of attestation.

4.8 Threat Modelling of Our Design for Provenance-based Attestation

The Microsoft Threat Modelling Tool is used for this work [67]. At the beginning of the threat modelling process, the tool resolves the target design using a Data Flow Diagram (DFD). A DFD will show all the elements involved in that design and the boundaries that represent the separation between system components or privilege level. This is followed by applying the STRIDE model to identify threat categories for every element in the DFD. STRIDE stands for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege. Chapter 6 contains further discussion on the Microsoft Threat Modelling Tool. A total of 39 potential threats were identified for our design shown in Figure 4.1. Appropriate mitigations for all the identified threats were worked out. We highlight some of the threats and the corresponding mitigation mechanisms in Table 4.1. The full threat modelling report is contained in Appendix B. These mitigation mechanisms can improve on the assurance of the trustworthiness conveyed by provenance-based attestation and can be addressed in future work.

4.9 A Protocol for Provenance-based Attestation

The protocol for provenance-based attestation describes how a provenance aware system (*AS-PAS*) stores a provenance record (*PR*) into the provenance record store (*AS-PRC*), how the attesting computer and trust evaluator communicate securely over the computer network, and how the trust evaluator verifies the authenticity of a provenance record. From threat modelling in Section 4.8, we know that provenance-based attestation is susceptible to the threat of spoofing and tampering. Therefore, we can make use of the digital signature and cryptographic functions offered by a hardware-based TPM 2.0 to mitigate this threat. We wanted the design to work for computers not equipped with the TPM and hence the steps involving the TPM can be replaced by other digital signature and cryptographic software, albeit with a different level of security. Listing 4.6 shows the sequence of events between the attesting provenance aware system, provenance record store and its TPM (*AS-TPM*).

At the moment the attesting computer *AS-PAS* produces a complete provenance record of an event, it will call the TPM of the attesting computer *AS-TPM* to perform a hash over the entire provenance record at line 2. From TPM 2.0 specification [92], the command to use is TPM2_SequenceComplete(). A hash digest *Digest_PR* is returned upon the successful completion of the command. Prior to this, *PAS* has obtained from the TPM a key pair for use with signing its provenance record. This key pair is derived from the storage primary seed of the TPM and it consists of a public verification key *PAS_SignKey_Pub* and a private sign-

Table 4.1: Threats and mitigations to provenance-based attestation.

S/N	Element	Type	Description	Mitigation
1	Provenance aware system	**S**	Attacker creates a false provenance record with the intention of misleading the trust evaluator.	The provenance aware system can digitally sign the provenance records. This digital signature will be verified by the trust evaluator. For high assurance, the TPM can be used to sign the provenance records. This process is only possible if the correct authorisation value is provided to the TPM.
2	Provenance aware system	**T**	Attacker alters the provenance records.	A signed hash digest of a provenance record is obtained before it is stored. For high assurance, the TPM can be used to generate this signed hash digest and produced a digital certificate to attest to this signed hash digest. This process is only possible if the correct authorisation value is provided to the TPM.
3	Attestation session between provenance aware system and trust evaluator	**T, I**	Attacker either tampers with the attestation session or reads the data.	Network cryptographic protocol such as the TLS can be used to secure the attestation session.

ing key *PAS_SignKey_Priv*. This private signing key in the TPM is loaded using the command TPM2_Load() at line 3. TPM2_Commit() then readies the signing key for use in a digital signature at line 4. Subsequently, at line 5, TPM2_Sign() takes in *DigestPR* of the provenance record and produce a digital signature using the committed signing key *PAA_PrivSignKey*. The outcome of these steps is that the provenance record is signed by the provenance aware system. The signing key pair is generated, based on attributes provided by *AS-PAS*, from the unique storage primary seed of the *AS-TPM*. Hence, this digital signature is uniquely linked to the *AS-PAS* and in turn linked to *AS-TPM*. Consequently, the threat of spoofing and tampering a provenance record is mitigated.

1 *AS-PAS* : produce provenance record PR
2 *AS-PAS* $\rightarrow$ *AS-TPM*: *Digest_PR* = TPM2_SequenceComplete(*PR*)
3 *AS-PAS* $\rightarrow$ *AS-TPM*: $Handle_{PAS\_SignKey\_Priv}$ =TPM2_Load(*PAS_SignKey_Priv*)
4 *AS-PAS* $\rightarrow$ *AS-TPM*: $ECC_{PAS\_SignKey\_Priv}$ = TPM2_Commit($Handle_{PAS\_SignKey\_Priv}$)
5 *AS-PAS* $\rightarrow$ *AS-TPM*: $Sign_{PR}$ = TPM2_Sign(*Digest_PR*, $ECC_{PAS\_SignKey\_Priv}$)
6 *AS-PAS* $\rightarrow$ *AS-PRC*: store *PR*, $Sign_{PR}$

Listing 4.6: Using TPM to generate digital signature of a provenance record.

At line 6, the provenance aware system will then store the provenance record *PR* and the digital signature *SignPR* into the provenance record store. To prevent an attacker from gathering information about the computer by reading the stored provenance record, the provenance record store can encrypt the provenance record *Encrypted_PR* using a cryptographic key *Crypt_Key*. This will be done using a symmetric cryptographic function such as the Advanced Encryption Standard [70].

During attestation over the computer network, the provenance record store will react to queries from the trust evaluator and reply with the relevant *Encrypted_PR*, *Crypt_Key*, *SignPR* and *PAS_SignKey_Pub*. This information exchange over the computer network can be secured using the transport layer security (TLS) [20]. The details of TLS implementation are widely available in the literature and hence are not covered in this thesis.

1 *TE* $\rightarrow$ *TE-TPM*: *Digest_PR* = TPM2_SequenceComplete(*PR*)
2 *TE* $\rightarrow$ *TE-TPM*: $Handle_{PAS\_SignKey\_Pub}$ = TPM2_Load(*PAS_SignKey_Pub*)
3 *TE* $\rightarrow$ *TE-TPM*: *Verify_PR* = TPM2_VerifySignature(*SignPR*, *Digest_PR*, *Handle* $_{PAS\_SignKey\_Pub}$)
4 If *Verify_PR* == success, proceed to evaluate *PR*

Listing 4.7: Using TPM to verify digital signature of a provenance record.

Listing 4.7 shows how the trust evaluator (*TE*) can verify the digital signature using its TPM (*TE-TPM*) . The trust evaluator will decrypt the provenance record using the cryptographic key. This will be done using an appropriate symmetric cryptographic function. To verify the integrity and authenticity of the provenance record, the trust evaluator will make use of its TPM. At line 1, the command TPM2_SequenceComplete() is called to obtain a hash digest of the provenance record. Then the verification key *PAS_SignKey_Pub* is loaded using TPM2_Load() at line 2. Next, at line 3, the command TPM2_VerifySignature() takes in *SignPR*, *Digest_PR*, $Handle_{PAS\_SignKey\_Pub}$. The outcome of the verification check is returned to the trust evaluator. If the check is successful, the trust evaluator will proceed to

assess the provenance record according to the rules. The details of this trust evaluation are described in the Section 4.6. Upon completion of trust evaluation, the trust evaluator will inform the provenance aware system about the outcome.

4.10 Related Works

Lyle et al. [47] noted that provenance and trusted computing could complement each other. They examined the requirements of these two domains and found that their principles and techniques could help to improve each other. Namiluko et al. [56] continued that work and proposed using provenance information of objects in a computing system for reasoning about trust properties. They extended the semantics of The Open Provenance Model (OPM) [53] to capture trust relations and discuss how these trust relations and provenance data can be used in the reasoning of trust properties. Our work differs in the intent and approach. We proposed a complete design on using provenance data for attestation. In Section 4.5, we discuss the collection of provenance data and present the use of the PROV data model. We made no extension to PROV semantics while Namiluko et al. modified the OPM data model. We further developed provenance-based attestation by proposing a grammar for specifying rules used for the trust evaluation of provenance records. In this aspect, our trust evaluation refers to rules that revolve around the dependencies among the target object and other elements and the attributes of these elements while Namiluko et al. checks for the authenticity of a software program or configuration by verifying its identity and integrity measurement value. Moreover, we conducted threat modelling on our design and propose a mitigation to the identified threats.

4.11 Summary

We have produced details on the collection of provenance data and leveraged the PROV data model to represent the provenance data. We have also developed a rule specification grammar and discussed how a provenance record can be evaluated during attestation. During the proof of concept, we have gained insights into how a provenance aware system can capture provenance data and by what method the provenance data in the provenance record can be assessed with rules to determine the trustworthiness of a target object. Thus, in this chapter, we show that the content of a provenance record can be used by another party to make a trust evaluation. We also show that rules can be formulated for the trust evaluation of a provenance record. Therefore, the design goals stated in Section 4.3 are met. In summary, we contribute a design for provenance-based attestation and a rule specification grammar for evaluating a provenance record. Future research can build upon this work, including those threat mitigation mechanisms that can improve on the assurance of the trustworthiness conveyed by provenance data.

Part II

Developing Trusted Systems

Chapter 5

An Ontology of a Computing Device Secured with Trusted Platform Module 2.0

5.1 Introduction

After examining trust properties, this part of the thesis moves on to deal with the development of a trusted system which uses the TPM 2.0 as a building block.

An ontology defines a common vocabulary for researchers who need to share information in a domain [57]. It includes machine-interpretable definitions of basic concepts in the domain and the relations among them. In recent years, the ontological approach has been used to build a common understanding between experts and developers of information security technologies. Karyada et. al. propose to use ontologies to capture and depict the knowledge of security experts [40]. In this way, developers can exploit security expertise in order to make design choices that will help them fulfill security requirements more effectively.

In this chapter, we will use an ontological approach to characterize security rooted in trusted hardware. The aim is to achieve a common understanding between security experts who established such technologies and the developers who make use of these technologies. As the field of trusted hardware covers numerous security hardware technologies, we framed our effort around the capabilities of a computing device secured with a Trusted Platform Module. This is to ensure meaningful results can be obtained with a reasonable amount of resources. The TPM is a device that is specified by the Trusted Computing Group (TCG). The TPM is designed to improve the trust in computing devices by offering certain functionalities such as cryptographic engine, secure storage, digital certification and trusted reporting of the identity and state of its host computing device. TPM 2.0 is the latest specification from TCG [92].

The outcome of our research is an ontology that characterizes the capabilities of a computing device secured with TPM 2.0. The main advantage of this ontology is that it can be used to describe, in a way that is easily comprehended and machine readable, the capabilities of such a computing device at various levels of abstraction. In addition, a developer can review the ontology by asking competency questions and these questions can be expressed by a query language.

The rest of this chapter is organized as follows. Section 5.2 gives the background to the use of ontologies. This is followed by Section 5.3 that describes the ontology development and explains the structure. Section 5.4 demonstrates how the ontology can be queried. Section 5.5 reviews the related works and the summary of this chapter is in Section 5.6.

5.2 Background

5.2.1 Characterizing the capabilities

On how to characterize the capabilities of a computing device secured with TPM 2.0, we referred to the technical models described in the National Institute of Standards and Technology Special Publication 800-33 [84]. This publication was discussed previously in Chapter 3. These models encompass information at several levels. They range from high level security objectives to low level specific technical details. For example, the low level technical primitive of cryptographic key management is described as enabling the capability of access control enforcement which in turn supports the security objective of confidentiality. The SP800-33 publication is intended to provide a description of the technical foundations that underlie security capabilities. Since our intentions are broadly aligned, we will frame our ontology of a computing device secured with TPM 2.0 on the structure of the technical models described in this publication.

5.2.2 Purpose of ontology

Ontology development involves giving explicit formal specifications of the terms in a domain and the relations among them [27]. Plainly, an ontology structures the information into classes and instances while their relationships are termed as properties. Classes refer to general concepts while an instance denotes a specific case. For example, cryptographic algorithm is a class and asymmetric key algorithm is a subclass. Then, RSA is a subclass of asymmetric key algorithm and RSA with key size 2048 is an instance of this subclass. In an ontology, a property refers to the semantic relationships among the classes and instances. For example, the ability of a computing device to produce a digital certificate is enabled by the asymmetric engine of TPM 2.0. Therefore, ”is enabled by” is the property that describes the semantic relationship between the production of digital certificate and asymmetric engine.

Hence, we can envisage that an ontology of the capabilities of a computing device secured with TPM 2.0 will have a class structure that depicts computer capabilities by classifying concepts under different classes. This class structure also allows us to describe the concepts at different abstract levels. In addition, the properties of these classes define the relationships among themselves. At the mean time, we consider this ontological approach as complementary to formal models such as [25]. An ontology based description provides benefits at a higher level by providing expressive, semantic meanings to relationships while a formal model acts at a finer level by giving unambiguous descriptions to these relationships. Furthermore, it is easier to obtain contextualized information from an ontology. This will be discussed in Section 5.5.

5.2.3 Web Ontology Language

The Web Ontology Language (OWL) was specified by the World Wide Web Consortium (W3C) in 2004 [93]. The language is based on the Resource Description Framework (RDF) [43]. RDF is a language for encoding information on Web pages

to assist computers searching for particular information. An RDF statement revolves around the triple model and it contains a subject, a predicate and an object. Consider the example RDF statement ”TPM is specified by TCG”. The RDF components for this statement are: the subject is ”TPM”, the predicate is the words ”is specified by” and the object is ”TCG”. This ability to describe the semantics of data enables RDF to support better discovery of resources and provides more meaningful description of content. OWL extends the RDF schema by providing a more expressive vocabulary and improving on the inference capabilities. A detailed description of OWL can be obtained from [93].

While there are other ontology languages such as the DARPA Agent Markup Language [33] and the Ontology Interface Layer [22], they have been superseded by OWL. Hence we reviewed OWL and found that the RDF triple model is especially useful for representing the classes and their relationships in an uncomplicated manner. In addition, OWL is designed to be compatible with the eXtensible Markup Language (XML) and this allows an OWL based ontology to be processed by the wide range of XML and RDF tools that are already available. This ability can be leveraged when we wish to query an ontology.

5.3 An Ontological Approach

5.3.1 Developing an ontology

We followed the methodology described in [57] to create this ontology. This methodology ensures that the developed ontology is fit for use in its intended application. In this work, we plan to use an ontology to characterize the capabilities of a computing device. This ontology is examined by a developer to determine if the design of a computing device meets certain trustworthiness requirements. Thus, this understanding is crucial to guiding the decision making during the ontology development.

The first step is to define the domain and scope in which this ontology will be used. We aim to represent, at different levels of abstraction, the description of the capabilities of a computing device secured with TPM 2.0. This covers trust primitives offered by TPM 2.0, capabilities and security objectives and their semantic relationships. A key activity in this step is the crafting of a set of competency questions. Competency questions refer to those questions the ontology should be able to answer when an entity wishes to understand the capabilities of the trustworthy computing device. These questions will later serve to verify if the developed ontology suits the intended application.

There are generally two types of competency questions. The first type refers to exploratory questions which the developer can ask to obtain a broad understanding of the computing device. The second type refers to specific questions on technical details. The following are examples of some of the possible competency questions:

Exploratory Questions

- What are the properties of the notion of confidentiality?
- Which TPM 2.0 capability enables the device capability of access control?

- Which TPM 2.0 subsystem does the TPM 2.0 capability of protected location use?

Specific Questions

- Is the TPM 2.0 capable of attestation using the asymmetric engine?
- Is AES one of the symmetric key algorithms used by this TPM 2.0?

The next step is to consider if an existing ontology can be reused. We studied the ontologies reviewed by Singh et al. [81] but we could not identify any ontology that focused on either trusted computing or TPM. We also undertook online searches and we could not find such an ontology. Thus, we believe that our endeavor is the first of its kind.

In step 3, we enumerated all the terms of TPM 2.0 that we considered to be relevant to the purpose of this ontology. This step was non-trivial as it took considerable time to acquire knowledge by understanding TPM 2.0 specifications [92], the book "A Practical Guide to TPM 2.0" [4] and the book "Trusted Computing Plaform, TPM 2.0 in context" [68]. This knowledge is supplemented by our experience from working on TPM 2.0 research projects and from our interaction with key personnel who have been involved in the development of TPM 2.0 specifications. Meanwhile, we investigated trust notions from Chapter 2. We also studied the security objectives listed in SP800-33. We believe that security and trust are not orthogonal. Trusted components are used to build secure systems. On the other hand, security properties are often evaluated as part of the process of establishing trust in a computing system. This association of trust and security is articulated in the Trusted Computer System Evaluation Criteria (Orange Book) of the United States of America Department of Defense [71]. Therefore, our ontology will include both security and trust notions. However, this step generates a long list of terms and we decided to put aside those that are not relevant to the purpose of this ontology. For example, there are terms such as power detection, execution engine, startup, shutdown and self-test which relate to the internal operations of TPM. These terms are not included in the ontology because they do not enable any capabilities of the computing device.

The remaining steps created the TPM 2.0 ontology. The steps are: defining classes and class hierarchy, defining class properties and their values, and creating instances. These steps are best illustrated together with the ontology and they are described and explained in the next sub section.

5.3.2 An Ontology of a Computing Device Secured with TPM 2.0

The development of classes, their hierarchy and properties are closely intertwined. We adopted a top down approach to develop the class hierarchy. The list of terms we obtained from step 3 were scrutinized and we defined the most general concepts. These general concepts are considered as classes. Then we zoomed in and identified those terms that belong to these classes. These terms were considered as subclasses. We reviewed the subclasses to check that they do not refer to the same concept. This avoided repetitions and ensure that there were no further groupings. For example, TPM 2.0 subsystem is a class and it has asymmetric engine as a

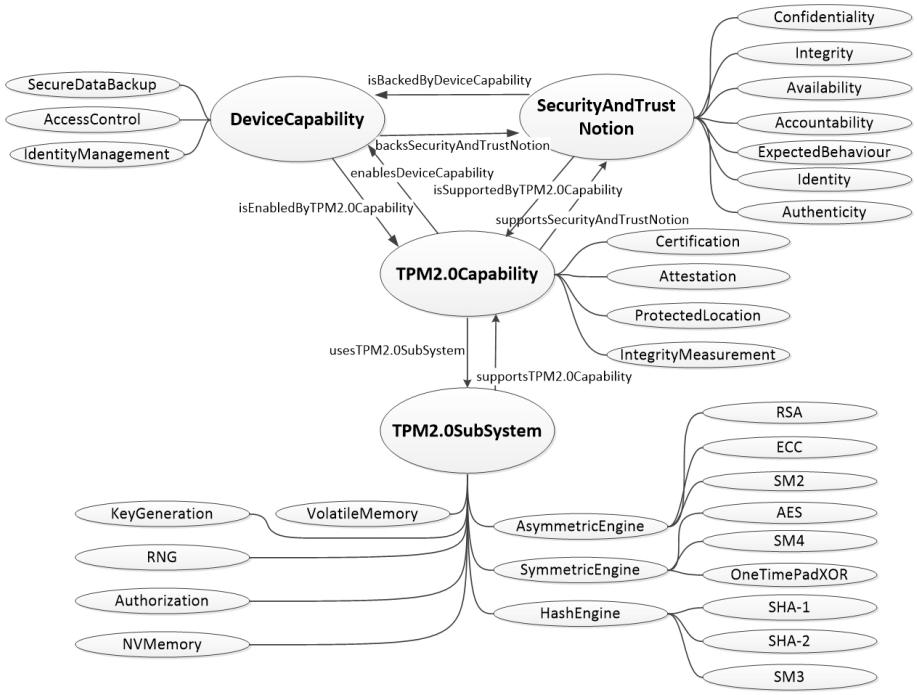


Figure 5.1: Graphical representation of the ontology based description.

subclass. Asymmetric engine further contains the algorithms RSA, ECC and SM2 as subclasses. Next, we leveraged our knowledge of TPM 2.0 and created properties that describe the semantic relationships between the classes. These steps were repeated to refine the ontology. After several exercises of sorting the terms into meaningful classes and ensuring that the ontology based description could meet our requirements, we derived 4 different classes.

The 4 classes are:

- TPM 2.0 Capability: this describes the capabilities a TPM 2.0 can offer to the host computing device. The capabilities are defined as subclasses of this class.
- TPM 2.0 Subsystem: this describes the subsystems of the TPM 2.0. Each of the subsystems is defined as a subclass.
- Security and Trust Notion: this is an amalgamation of user level trust notions and security objectives. Each of the notions is defined as a subclass of this class.
- Device Capability: this describes at the abstract level what trusted functions are offered in a particular computing device. Some examples of trusted functions are defined as subclasses in this ontology.

We use OWL as the language for our ontology and this ontology is created using Protege [1]. Protege is an open source platform that provides a suite of tools to construct ontologies. We chose Protege because it supports RDF and XML format

[1] http://protege.stanford.edu/

and the ontology can be queried using SPARQL, an RDF data access language [69]. Figure 5.1 shows the graphical representation of these classes. The full ontology in OWL format is given in Appendix C of this thesis.

The core class in this ontology is the *TPM2.0Capability class*. This class serves as the linkage between low level trust primitives described in the *TPM2.0SubSystem* class and high level device capability descriptions and user notions. It mirrors the support services described in SP800-33 and characterizes the capabilities of TPM 2.0 which enable many trusted functions of the computing device. It consists of the subclasses *Certification, Attestation, ProtectedLocation and IntegrityMeasurement*. These subclasses were mainly referenced from TPM 2.0 specifications while additional information was gathered from [4] and [68]. We consider them as subclasses because they are general concepts of the capabilities offered by TPM 2.0.

The *TPM2.0Capability* class is supported by primitives that are described in the *TPM2.0SubSystem* class. There are 11 subclasses in *TPM2.0SubSystem* class and they are *AsymmetricEngine, SymmetricEngine, HashEngine, KeyGeneration, RNG, Authorization, NVMemory, VolatileMemory, ExecutionEngine, PowerDetection and Management*. We have also defined another level of subclasses to describe the algorithms used in the subclass of *AsymmetricEngine, SymmetricEngine* and *HashEngine*. These subclasses were referenced from TPM 2.0 specifications and supplementary information was gathered from [4] and [68]. The motive for *TPM2.0SubSystem* class is to depict the primitives that underlie TPM 2.0 capabilities. Thus, the *TPM2.0Capability* class is mapped to the *TPM2.0SubSystem* class by the property *usesTPM2.0subsystem*. We can make use of the property value restriction feature of OWL to indicate that a particular subclass of *TPM2.0Capability* class is only mapped to some subclass of *TPM2.0SubSystem*. We also define *supportsTPM2.0Capability* which is an inverse to the property *usesTPM2.0subsystem*. We have a two way mapping because it will be useful when we query the ontology and wish to transverse through the classes to uncover more information. An example of how the *TPM2.0Capability* class is mapped to *TPM2.0SubSystem* class is as follows: The encryption of a protected location uses a symmetric key and this symmetric key is generated from a seed. Meanwhile, the seed is stored in the non-volatile memory of TPM 2.0. Thus, *ProtectedLocation* is defined to have *SymmetricEngine, KeyGeneration* and *NVMemory* as its *useTPM2.0subsystem* property values.

The *TPM2.0Capability* class is mapped to the *SecurityAndTrustNotion* class to illustrate how TPM 2.0 capability supports the high level security and trust notions. This mapping also serves as the bridge to low level trust primitives. The *SecurityAndTrustNotion* class contains 7 subclasses and they are *Confidentiality, Integrity, Availability, Accountability, ExpectedBehaviour, Identity and Authenticity*. These subclasses are referenced from the security objectives of SP800-33 and the objective trust notions mentioned in Chapter 2. Subjective trust notions are not considered in this ontology because their perception can defer from person to person. We consider these objective trust notions as subclasses and they can be populated later with instances that identify specific references. For example, secrecy of user password can be an instance of the subclass *Confidentiality*. The *TPM2.0Capability* class is mapped to *SecurityAndTrustNotion* class via the property *supportsSecurityAndTrustNotion*.

Conversely, we also define *isSupportedByTPM2.0Capability* which is an inverse

of the property *supportsSecurityAndTrustNotion*. As an example, assume that the high level user trust notion of *ExpectedBehaviour* is desired and, from the ontology based description, we know that the *Attestation* and *IntegrityMeasurement* subclass of *TPM2.0Capability* support this notion via the property *isSupportedByTPM2.0 Capability*. As we trace the ontology based description further, we can see that *Attestation* is linked via the property of *useTPM2.0subsystem* to *AsymmetricEngine* of *TPM2.0SubSystem* class while *IntegrityMeasurement* is linked via the same property to *HashEngine* and *VolatileMemory*.

Lastly, we define the *DeviceCapability* class to demonstrate how low level trust primitives can be mapped to high level device functions. The *DeviceCapability* class is mapped to the *TPM2.0Capability* class by the property *isEnabledByTPM2.0-Capability*. We also define *supportsTrustNotion* which is an inverse of the property *isEnabledByTPM2.0Capability*. In this ontology based description, we defined 3 examples of device functions that are described in [4]. They are *SecureDataBackup, AccessControl* and *IdentityManagement*. These subclasses are used as examples here and hence they are not fixed. The user of this ontology based description can define more subclasses to describe other functions. As an example, in access control, we want different users to have separate levels of access to a database. This function can be enabled by leveraging the protected location function of TPM 2.0 which in turn makes use of the authorization subsystem and symmetric key cryptographic engine. Hence, from the ontology based description, *AccessControl* is mapped to *ProtectedLocation* of the *TPM2.0Capability* class via the *isEnabledByTPM2.0Capability* property. In turn, *ProtectedLocation* is mapped to *Authorization* and *SymmetricEngine* of *TPM2.0SubSystem* class via the *usesTPM2.0SubSystem* property.

In this section, we introduced the ontology and characterized the constituent classes. We also gave examples of how the ontology can be used to describe the trustworthiness of a computing device by providing a map of low level trust primitives to high level notions and capabilities of computing device. This ontology will serve as useful foundation whereby experts can further populate the ontology with instances that describe more specific implementation and configuration information.

5.4 Querying

This section will address how a developer can query the ontology based description to uncover its content. We studied the ontology and its OWL/RDF format and propose to translate the competency questions in Section 5.3.1 into SPARQL queries. There are two types of questions. The intent of exploratory questions is to uncover information by checking for the content of the components in a RDF triple. The content can be used to infer certain information about the computing device. For example, we refer to the competency question on the properties of confidentiality. Listing 5.1 shows the description of the *Confidentiality* class in RDF/XML format. If we wish to know all about its properties and values, a query using the SPARQL function of SELECT and WHERE can be issued. Figure 5.2 shows the query and Table 5.1 shows the result.

```
SELECT ?property ?value
WHERE
{tpm2.0:Confidentiality rdfs:subClassOf ?object .
?object owl:onProperty ?property .
?object owl:someValuesFrom ?value }
```

Figure 5.2: SPARQL query for the *Confidentiality* class.

Table 5.1: Result for SPARQL query on the *Confidentiality* class.

S/N	Property	Value
1	isSupportedByTPM2.0Capability	ProtectedLocation
2	isBackedByDeviceCapability	SecureDataBackup
3	isBackedByDeviceCapability	AccessControl

```
1  <owl:Class rdf:about="&tpm2.0;Confidentiality">
2  <rdfs:subClassOf rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
3  <rdfs:subClassOf><owl:Restriction>
4  <owl:onProperty rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
5  <owl:someValuesFrom rdf:resource="&tpm2.0;ProtectedLocation"/>
6  </owl:Restriction></rdfs:subClassOf>
7  <rdfs:subClassOf><owl:Restriction>
8  <owl:onProperty rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
9  <owl:someValuesFrom rdf:resource="&tpm2.0;AccessControl"/>
10 </owl:Restriction></rdfs:subClassOf>
11 <rdfs:subClassOf><owl:Restriction>
12 <owl:onProperty rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
13 <owl:someValuesFrom rdf:resource="&tpm2.0;SecureDataBackup"/>
14 </owl:Restriction></rdfs:subClassOf>
15 </owl:Class>
```

Listing 5.1: Description of the *Confidentiality* class in RDF/XML format.

In another example, we refer to the competency question on which TPM 2.0 capability enables the device capability of access control. Listing 5.2 shows the OWL/RDF description of *AccessControl* class.

```
<owl:Class rdf:about="&tpm2.0;AccessControl">
<rdfs:subClassOf rdf:resource="&tpm2.0;DeviceCapability"/>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;backsTrustNotion"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;Accountability"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;backsTrustNotion"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;Integrity"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;backsTrustNotion"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;Confidentiality"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;backsTrustNotion"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;Availability"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;isEnabledByTPM2.0Capability"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;ProtectedLocation"/>
</owl:Restriction></rdfs:subClassOf>
</owl:Class>
```

Listing 5.2: Description of the *AccessControl* class in RDF/XML format.

For this query, we can make use of the FILTER function of SPARQL. Figure 5.3 shows the query and Table 5.2 shows the result. Meanwhile, the same query can be altered to obtain answer to the competency question of which TPM 2.0 subsystem does the TPM 2.0 capability of protected location uses.

```
SELECT ?property ?value
WHERE
{tpm2.0:AccessControl rdfs:subClassOf ?object .
?object owl:onProperty ?property .
?object owl:someValuesFrom ?value
FILTER regex (str(?property), "isEnabledByTPM2.0Capability")}
```

Figure 5.3: SPARQL query for the *AccessControl* class.

Table 5.2: Result for SPARQL query on the *AccessControl* class.

S/N	Property	Value
1	isEnabledByTPM2.0Capability	ProtectedLocation

```
ASK { tpm2.0:Attestation rdfs:subClassOf ?object .
?object owl:onProperty tpm2.0:usesTPM2.0SubSystem .
?object owl:someValuesFrom tpm2.0:AsymmetricEngine }
```

Figure 5.4: SPARQL query for the *Attestation* class.

A specific competency question is concerned with the presence of RDF triples. For example, one can check the ontology to determine if the RDF triple (Attestation, usesTPM2.0SubSystem, AsymmetricEngine) exists. This example refers to the competency question of asking if the TPM 2.0 capability of attestation is using the asymmetric engine. The existence of this triple indicates that the TPM 2.0 capability of attestation uses the asymmetric engine of TPM 2.0 subsystem. Listing 5.3 shows the OWL/RDF format of the description for the Attestation class.

```
1  <owl:Class rdf:about="&tpm2.0;Attestation">
2  <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0Capability"/>
3  <rdfs:subClassOf><owl:Restriction>
4  <owl:onProperty rdf:resource="&tpm2.0;supportsTrustNotion"/>
5  <owl:someValuesFrom rdf:resource="&tpm2.0;ExpectedBehaviour"/>
6  </owl:Restriction></rdfs:subClassOf>
7  <rdfs:subClassOf><owl:Restriction>
8  <owl:onProperty rdf:resource="&tpm2.0;supportsTrustNotion"/>
9  <owl:someValuesFrom rdf:resource="&tpm2.0;Identity"/>
10 </owl:Restriction></rdfs:subClassOf>
11 <rdfs:subClassOf><owl:Restriction>
12 <owl:onProperty rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
13 <owl:someValuesFrom rdf:resource="&tpm2.0;Authorization"/>
14 </owl:Restriction></rdfs:subClassOf>
15 <rdfs:subClassOf><owl:Restriction>
16 <owl:onProperty rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
17 <owl:someValuesFrom rdf:resource="&tpm2.0;NVMemory"/>
18 </owl:Restriction></rdfs:subClassOf>
19 <rdfs:subClassOf><owl:Restriction>
```

```
<owl:onProperty rdf:resource="&tpm2.0;enablesDeviceCapability"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;IdentityManagement"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;AsymmetricEngine"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;VolatileMemory"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;KeyGeneration"/>
</owl:Restriction></rdfs:subClassOf>
<rdfs:subClassOf><owl:Restriction>
<owl:onProperty rdf:resource="&tpm2.0;supportsTrustNotion"/>
<owl:someValuesFrom rdf:resource="&tpm2.0;Authenticity"/>
</owl:Restriction></rdfs:subClassOf>
</owl:Class>
```

Listing 5.3: Description of the *Attestation* class in RDF/XML format.

We can use the ASK command to query the ontology for a specific triple and the command will return true or false depending on whether the specific triple can be found. Figure 5.4 shows the SPARQL query. The result for this query is "True". Meanwhile, the same query can be modified for the competency question "Is AES one of the symmetric key algorithms used by this TPM 2.0?"

In this section, we only discuss the queries that can be used to help to answer the example competency questions posted in Section 5.3.1. This discussion will serve as an introduction and there can be many other questions that the ontology can answer using the powerful SPARQL tool. In the meantime, the ability of the ontology described in Figure 5.1 to answer these questions validate that its structure and content are suitable for use by a developer to uncover information about the design of a computing device secured by TPM 2.0.

5.5 Related Works

Kim et al. noted that annotation with security related metadata enables discovery of resources that meet security requirements [41]. They presented the Naval Research Laboratory (NRL) Security Ontology which focuses on the annotation of security mechanisms, protocols, algorithms, credentials and objectives. The NRL Security Ontology consists of the core class of Security Concept. It has subclasses of Security Protocol, Security Mechanism and Security Policy. The Security Concept class is defined to support the Security Objective class. The other classes of the NRL Security Ontology include Security Algorithms, Security Assurance, Credentials, Service Security, Agent Security and Information Object. The authors explained that the classes of Service Security, Agent Security and Information Object

classes are extensions of the DAML Security Ontology [18]. On the other hand, the Credentials, Security Algorithm and Security Assurance classes provide values for properties defined for concepts in the Security Concept class. As the authors applied this ontology to a Web Service Oriented Architecture, the Information Object class was added to allow for the annotation of web service inputs and outputs. In the paper, the authors also described an algorithm to perform matching of security capabilities to security requirements. Although our ontology has the same intent and approach as the NRL Security Ontology, we differ in the domain of the ontology. Our domain is to describe the capabilities of a computing device secured by TPM 2.0. We also explicitly explain how we can use SPARQL to query the ontology for answers to the competency questions. On the other hand, the NRL Security Ontology described how the matching algorithm works but was not clear on how the algorithm can be implemented.

5.6 Summary

We have provided details on how an ontology of a computing device secured with TPM 2.0 were developed and explained the contents and arrangements of the classes and their properties. We have also given sample competency questions that should be answered by this ontology and use SPARQL to demonstrate how to obtain answers to the competency questions. Thus, from this work, we have gained insights into how an ontology can be used as a standard vocabulary for experts to share information on TPM 2.0 with developers of trusted systems.

Chapter 6

Threat Model of a Scenario based on Trusted Platform Module 2.0 Specification

6.1 Introduction

Protection offered by hardware security mechanisms, such as the TPM 2.0, can significantly strengthen a trusted system. However, it is naive to implement TPM 2.0 without identifying threats to its use scenarios and apply appropriate mitigations. There are several papers presented in the past discussing attacks on the TPM 1.2 specification, for example, [11], [12], [9], [28]. However, these works look at the older TPM 1.2. Moreover, these works focused on examining TPM protocols, identifying weakness, and suggesting solutions to the problems. No threat model for use scenarios of TPM was developed. Thus, in this chapter, we study the use of threat modelling for the purpose of identifying threats to TPM 2.0 use scenarios and recommending appropriate mitigations.

The rest of this chapter is organized as follows. Section 6.2 explains the threat modelling methodology. In Section 6.3, we describe the scenario for the threat model and it is followed by threat identification and mitigations in Section 6.4. Related works are discussed in Section 6.5 and the summary of this chapter is in Section 6.6.

6.2 Threat Modelling

6.2.1 Background

Poor design of software internal code and interfaces during the design phase of a computing device will eventually lead to vulnerabilities that could be exploited by an attacker [86]. It is recommended to fix an issue during design phase where it involves reworking a design rather than later in the production phase where it requires significant reengineering and patch releases [19]. Microsoft suggested to perform threat modelling early in the development life cycle because it can reveal weaknesses that may require significant changes to the product [86].

Threat modeling is the process of enumerating and risk-rating malicious agents, their attacks, and those attacks' possible impacts on a system's assets [83]. Threat modelling happens either at the organization level or at the system level. In this chapter, we study threat modelling at the system level. This process employs the following steps:

1. Identify security objectives.

Element Type	Threat Types					
	S	*T*	*R*	*I*	*D*	*E*
External Entity	X		X			
Process	X	X	X	X	X	X
Data Storage		X	X	X	X	
Data Flow		X	X		X	

Figure 6.1: STRIDE-per-element matrix from [34].

2. Create an application overview.
3. Decompose the application.
4. Identify threats.
5. Identify vulnerabilities.
6. Work out mitigations.

Microsoft has made public numerous articles to promote its threat modelling methodology [78], [35], [34], [67]. At the beginning of their threat modelling process, the tool resolves the target scenario using a Data Flow Diagram (DFD). A DFD will show all the elements involved in that scenario. An element can be an external entity, a process, a data store or a data flow. A boundary that represents the separation between system components or privilege level will then be defined. This is followed by applying the STRIDE model to identify threat categories for every element in the DFD. STRIDE stands for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege. Only certain threat categories can apply to certain elements [34]. Please see Figure 6.1.

The tool will automatically generate the threat categories for each element based on Figure 6.1 but each threat category has to be analysed manually. The tool guides the identification of specific threats by providing a set of questions. For every identified threat, an appropriate mitigation should be worked out. Before the threat model report can be generated, additional information on assumptions, external dependencies and security notes can be entered into the tool. It is important to note that the threat model report is a live document and it should be constantly updated whenever a new threat is detected or there is a configuration change to the target scenario.

Besides Microsoft's secure development life cycle threat modelling tool, there is an array of threat modelling frameworks and tools, such as OCTAVE from Carnegie Mellon University's Software Engineering Institute [10] and the open source TRIKE [76]. The OCTAVE threat modelling method is performed at the organization level and hence it is not suitable for this study on system level threat modelling.

On the other hand, TRIKE is used to build threat models in order to support the security auditing process from a risk management perspective. It uses spreadsheets to describe different aspects of the system being modelled. This threat modelling

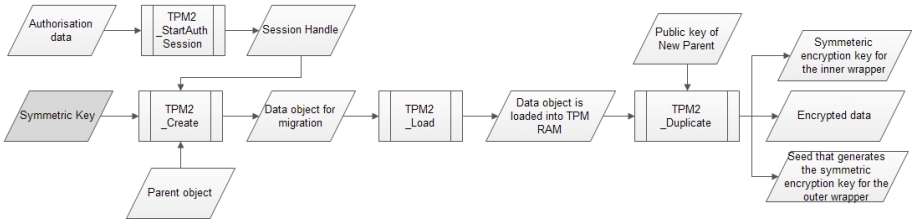


Figure 6.2: To encrypt symmetric key for group share.

process begins with requirement analysis. This step involves the identification of actors, assets and how actors interact with assets. This information is then used to construct an actor-asset-action matrix. The next step requires the mapping of this matrix to the DFD of the target scenario. Thereafter, threats are generated and appropriate mitigations are worked out.

TRIKE shares similarities with the Microsoft threat modelling process. However, key differences include the number of threat classifications (2 for TRIKE and 6 for STRIDE), less support for threat mitigation and the fact that TRIKE has weaker architecture representation of the target scenario. In addition, we find that the Microsoft threat modelling tool is better developed, easier to use and has more refined user interface. Thus, we decided to use the Microsoft threat modelling tool in this work.

6.3 Description of Scenario

We refer to the ontology described in Chapter 5 of this thesis and the TPM 2.0 specification [92] when crafting this scenario. In this simplified scenario, TPM 2.0 is used to encrypt the cryptographic key used for encrypting data for sharing with a group. This allows the key to be securely exchanged. This scenario is selected because it uses TPM's shielded storage feature and is applicable to a web application situation where certain sensitive web data has to be securely shared with other users over a computer network. The scenario illustrated in Figure 6.2 describes how a symmetric key used for encrypting data is shared using TPM's key duplication function. References to TPM commands from Chapter 3 of the TPM 2.0 specification are made at key points of this process. It is noted that TPM 2.0 commands are different from those of TPM 1.2.

In Figure 6.2, TPM2_Create is used to package the key into a TPM object. But before the command can be executed, an authorisation session for the use of a parent object to create the child TPM object has to be started. Upon successful authorisation, TPM2_Create command will execute and produce a data object that contains the key. This data object will have a flag setting indicating that it can be duplicated. In addition, a user can specify an authorisation policy to control access to this data object. The next step is to load this data object into the TPM RAM using the command TPM2_Load. This command will return a handle to the key object. The final command to run is TPM2_Duplicate whereby this data object is repackaged and encrypted. The output from TPM2_Duplicate is the encrypted duplicated object, the symmetric encryption key used to encrypt the inner wrapper and a seed that

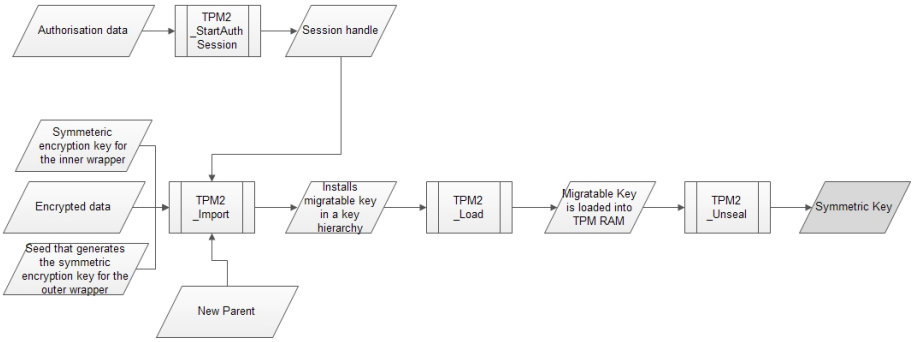


Figure 6.3: To recover symmetric key for group share.

generates the symmetric encryption key for the outer wrapper. The confidentiality of the seed value is protected by a public key provided by the destination TPM. These outputs can be transferred to the destination TPM using a mechanism that protects the confidentiality and integrity of the duplicated object and checks the authenticity and authorisation of the destination TPM.

At the destination TPM, the reverse is carried out. Referring to Figure 6.3, TPM2_Import is used to transfer the duplicated object into the destination TPM. An authorisation session for the use of the new parent object is started. Upon successful authorisation, the command will execute and the duplicated object is decrypted. To protect the confidentiality of the key object, it is encrypted with an encryption key derived from the new parent. This key object is then loaded into the TPM RAM using the command TPM2_Load. A handle to the loaded key object is returned to the user. To obtain the symmetric key, the authorisation data and key object handle are provided to the command TPM2_Unseal. When this command executes successfully, the symmetric key is presented to the user.

6.4 Threat Identification and Mitigation

Using Microsoft's secure development lifecycle threat modelling tool, two DFDs were drawn to represent the scenario of encrypting and decrypting the symmetric key for group share. The DFDs are shown in Figure 6.4.

This tool analyzed the two DFDs individually and threat categories for every element were generated. A total of 101 potential threats were identified for the process of encrypting the symmetric key for group share while a total of 96 potential threats were identified for the decrypting process. The data flow between the processes and the TPM RAM are not accessible externally and hence they were not analyzed (grey coloured lines). Appropriate mitigations for all the identified threats were worked out. TPM 1.2 attacks [11], [12], [9], [28] identified in earlier studies could not be applied directly to this threat model as the protocols and commands for TPM 2.0 have been changed.

The threat modelling report is given in Appendix D but we will discuss some of the more critical ones in Table 6.1.

S/N	Element	Type	Description	Mitigation
1	TPM2_Import	**S**	Attacker attempts to load a duplicated key object that is not generated by a TPM.	The source TPM can insert a unique identifying value into the key object when using TPM2_Create. The destination TPM will verify the authencity of the key object by inspecting this identifying value.
2	TPM2_ StartAuthSession	**I**	The cryptographic protection for the authorized sessions can be weakened if the nonce and salt value used in the generation of the session key have low entropy.	The method used by the software application to generate the nonce and salt value has to meet security requirements, for example NIST SP 800-90A. An alternative method is to use TPM's random number generator (RNG) to provide these values. However, TPM's RNG has to meet security requirements as well.

3	Key object (TPM2_Create to User Application)	**I**	The sensitive part of the key object is symmetrically encrypted using a key derived from the parent object. A random value is included in the process as an initialization vector (IV). When an object is created for duplication, the IV is set to zero. The key objects can be susceptible to cryptographic analysis if the parent object is reused multiple times.	The user application has to avoid reusing the parent object multiple times when creating an object for duplication.
4	TPM2_Create	**I**	User denies executing this command.	TPM will have to rely on the TCB to keep a log of the commands performed on TPM. The availability of a log is crucial to forensic investigation in the event of a security incident. An example of a guideline for the security management of the log will be NIST SP 800-92.

Table 6.1: Threats and mitigations to use scenario.

Since TPM's design objectives do not include protection from physical attacks, this thesis will not dwell on this threat but a user should be aware of the types of physical attack [61], [87] and take appropriate mitigations.

6.5 Related Works

Möckel and Abdallah discussed the threat modelling process and its usefulness to the design of an electronic banking application [52]. They also used the Microsoft threat modelling tool. However, we went further and discussed about other threat modelling tools such as OCTAVE and TRIKE and explained why we chose to use

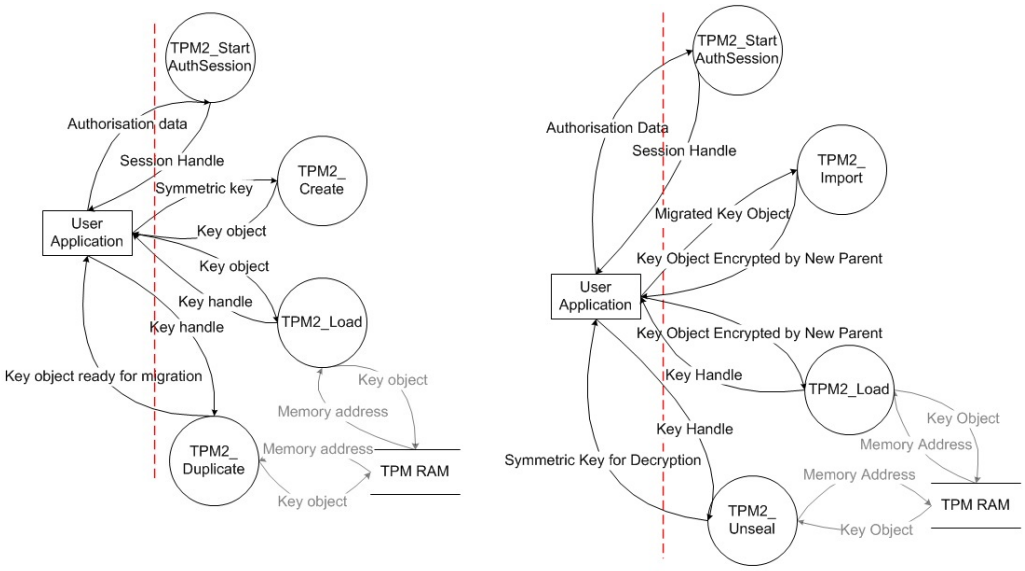


Figure 6.4: DFD for encrypting symmetric key (left) and for recovering symmetric key (right).

the Microsoft tool. Moreover, they used the threat modelling tool on the abstract design of an application while we applied threat modelling to the use scenario of a trusted firmware.

6.6 Summary

In this chapter, we have developed a use scenario for TPM 2.0 and studied the use of threat modelling for the purpose of identifying threats and mitigations to TPM 2.0 use scenarios. The scenario is on using a TPM to share a symmetric cryptographic key and the threat model of this scenario is produced. The usage of threat modelling allows us to recommend improvements to the security of TPM 2.0 use scenarios.

Chapter 7

Para-Virtualizing the Trusted Platform Module: An Enterprise Framework Based on Version 2.0 Specification

7.1 Introduction

We now consider how TPM 2.0, as a building block for trust properties, can be used in a modern system. Virtualization is a fundamental technology that is widely used in Enterprise IT infrastructures. Users of virtualization technology need some level of assurance about the expected behavior of a virtual machine (VM) and its ability to protect confidential information from unauthorized disclosure. The TPM specified by the Trusted Computing Group (TCG) offers security properties that can be leveraged by the users of virtualization technology to increase the protection of the system and data from cyber security threats [77].

However, the TPM was originally designed for use with a computing system in a one to one relationship. In a virtualized system, the design will require enhancements to the TPM in order for it to work in an environment where a computer hardware platform hosts several VMs. There are generally two types of techniques to enable a TPM hardware chip to support multiple VMs. The first type is full virtualization of the TPM which is exemplified by the work of Perez et al. [64]. In that paper, the authors described the creation of software virtual TPM instances contained in a privileged VM. Each virtual TPM will support a unique VM. This design is aligned to the virtual TPM framework proposed in TCG's Virtualized Trusted Platform Architecture Specification [88] and Open Trusted Computing's VTPM Architecture [59]. Although the designs often extend the chain of trust from the TPM hardware chip to the virtual TPM, the security protection for confidential data provided by the TPM's hardware based protected storage is not offered. A probable reason could be that these designs are based on the older TPM 1.2 specification and the limited amount of TPM memory is unable to support the requirements of the virtualized environment. In addition, the long chain of trust from the TPM hardware chip to the virtual TPM can be fragile as the attack surface is now larger compared to a non-virtualized implementation.

This chapter will analyze the other type of technique which is para-virtualizing the TPM. England and Loeser wrote that TPM para-virtualization refers to the method of mediating guest VM access to hardware TPM using a software component [21]. The design will require no change to most TPM functionality but some aspects of the device interface may change. A major advantage offered by this technique is that the access control to TPM's protected storage resides in the hardware chip and this feature is desired by organizations that require hardware

based security; for example, government Enterprise IT. Moreover, the chain of trust is now shorter as the VM can access the TPM hardware chip in a more direct manner. However, in a para-virtualization design, the use of the resources belonging to a single TPM by multiple VMs has to be managed to ensure fair sharing and prevent cross-interference. On the other hand, the TPM has to provide sufficient resources to support the operation of several VMs. With the advent of the newer TPM 2.0 specification, it is timely to examine if the newer specification can better support para-virtualization requirements. We will introduce a para-virtualization framework that leverage new capabilities offered by TPM 2.0. The framework will also address the challenges for achieving TPM para-virtualization in Enterprise IT, for example, backup and migration.

This chapter presents a theoretical research rather than a presentation of an actual implementation. Section 7.2 contains a quick study of the new TPM 2.0 specification from the TCG and Section 7.3 analyzes the state of the art regarding para-virtualizing the TPM. With this background knowledge, Section 7.4 will then examine the extent to which TPM 2.0 core functions are suitable for para-virtualizing. This is followed by design requirements in Section 7.5. Section 7.6 describes a theoretical framework for para-virtualizing TPM 2.0 in the context of an Enterprise IT infrastructure. Section 7.7 revisits the design requirements and the summary of this chapter is in Section 7.8.

7.2 More About TPM 2.0

The Trusted Computing Group (TCG) wrote in the Trusted Platform Module (TPM) version 2.0 specification [92] that trust conveys an expectation of behaviour from the computer system. In other words, a user can trust a computer if it always behaves as it is intended to. The assessment of trust always begins from some baseline, or "root of trust". In a computing platform, the three roots of trust for measurement, storage and reporting provide the minimum functionality required to describe the attributes that contribute towards its trustworthiness. The TPM and supporting components aim to provide these three roots of trust.

TPM 2.0 is the latest specification from the TCG and it replaces the previous TPM 1.2 specification. The changes and enhancements to TPM 2.0 compared to the previous TPM version include: support for additional cryptographic algorithms, enhancements to the availability of the TPM to applications, enhanced authorization mechanisms, simplified TPM management, and additional capabilities to enhance the security of platform services.

7.2.1 Architecture

TPM 2.0 is designed to be a self-contained computing device. This allows the TPM device to be trusted to carry out computations without relying on external computing resources. The following are short descriptions of the subsystems in a TPM 2.0 device while a detailed explanation can be obtained from the TPM 2.0 specification and the ontology described in Appendix C.

I/O Buffer This component enables the host computing system to communicate with the TPM. It can be a shared memory. Data to be processed by the TPM will be

validated at this point.

Cryptography Subsystem The cryptographic engine supports commonly used cryptographic functions like hashing, asymmetric operations such as digital signature and key exchange, symmetric encryption, random number generator and key derivation function. These cryptographic functions can be used by the other TPM components or the host computer.

Authorization Subsystem Before a TPM command is executed, this subsystem checks that proper authorization data has been given by the calling application.

Volatile Memory This memory holds transient TPM data, including Platform Configuration Registers (PCRs), data objects and session data. A PCR contains the integrity measurements of critical components in the host computer. A data object can either be a cryptographic key or other data. The TPM uses sessions to manage the execution of series of commands.

Non-Volatile (NV) Memory This memory is used to store persistent TPM data that includes the platform seed, endorsement seed, storage seed and monotonic counter. Additional PCR banks can be created in this memory.

Management Subsystem This subsystem oversees the operation of the various TPM states. Basic TPM states include power-off, initialization, start up, shut down, self-test, failure and field upgrade.

Execution Engine This firmware contains the program instructions and data structures that are required to run a TPM command. These program instructions and data structures cannot be altered by the host computing platform. In the event of a firmware upgrade, there are security mechanisms to ensure that the update is authorized and the new firmware is checked for authenticity and integrity.

7.2.2 Core Functions

A Trusted Computing Base (TCB) can be a BIOS, Virtual Machine Monitor (VMM) or operating system that has proved to be highly secure and hence trustworthy. When a TCB is made to work together with a TPM, they can offer the capabilities of integrity measurement and reporting, protected data storage, certification and attestation and authentication. In integrity measurement, a hash function is performed by the BIOS on the first software component that is started when the computer powers up. The hash function will produce a digest of that software component. If that software component is altered, the digest will be different from the one obtained when the software component was first measured. This digest can be stored in the PCR located in either the volatile or non-volatile memory of the TPM. TrustedGRUB [1] is an application that implements this integrity measurement at system start. A TPM can have an authenticity certificate from the manufacturer and this feature is used in conjunction with the integrity measurement to report the "trustworthiness state" of a computing platform.

A unique feature of the TPM is the use of primary seeds to generate hierarchies of keys for use in cryptographic functions. The intention of this feature is to provide the flexibility to support different types of cryptographic functions without increasing the storage memory requirement. To establish trust in a key derived from a TPM primary seed, the TPM can produce a digital certificate indicating that

[1] http://www.trust.rub.de/projects/trustedgrub/

the processes used for creating and protecting the key meet the necessary security requirements. During attestation, a TPM can vouch for the authenticity and properties of either the host computing platform, a piece of software or a cryptographic key.

TPM non-volatile memory is typically used to store cryptographic keys that protect sensitive data. In this method, the sensitive data is encrypted with a cryptographic key derived from a root key inside the TPM chip. This cryptographic key is usually stored into the non-volatile memory of the TPM chip. To read the sensitive data, the user has to provide an authorization data to the TPM chip and only upon successful authorization will the cryptographic key be released from the TPM chip. This is known as protected storage.

7.3 State of the Art for Para-Virtualizing the TPM

The following sub-sections will give brief descriptions of two projects on para-virtualizing TPMs. It is important to note that these two projects are based on the older TPM 1.2 specification.

7.3.1 Para-Virtualized TPM Sharing

In [21], the authors describe a para-virtualization design that allows a VMM to time share a TPM among its VMs. The concept of associating a TPM context to a particular VM is proposed. A TPM context will contain the important data that defines a TPM state, for example, keys and sessions. When a particular VM wishes to use the physical TPM, the associated TPM context is loaded into that physical TPM. The loaded TPM context can be saved and cleared from the physical TPM to allow another TPM context to be loaded when required. TPM contexts are saved in the hypervisor. As a result, the design is able to support most TPM applications.

To implement this design, the authors located the TPM para-virtualization management software in the hypervisor. Figure 7.1 gives a high level view of this design.

The TPM para-virtualization management software contains the following components:

Scheduler Schedules shared access to the physical TPM.

Command Blocking Filters TPM commands based on a pre-determined list of allowed commands. This is to ensure the safe operation of the TPM by disallowing applications in the VM from executing certain TPM commands.

Virtual PCRs Each VM is assigned a set of virtual PCRs and they are managed by the hypervisor.

Context Manager This component inspects every TPM command and loads the associated context into the physical TPM so that the VM can only access its own TPM resources.

Resource Virtualization Certain limited TPM resources are virtualized, for example, key slots, authorization sessions and transport sessions. These virtualized TPM resources are tied to a context that is provided by the context manager.

The use of virtual PCRs to store integrity measurements of the VM is not desired by organizations that require hardware based protected storage. Furthermore, the

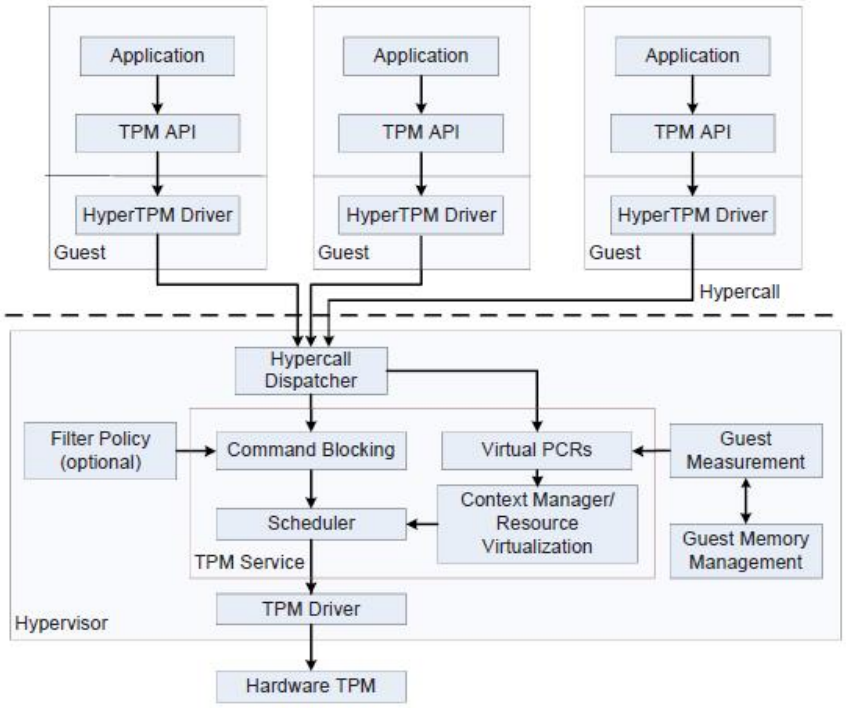


Figure 7.1: Architecture for para-virtualizing TPM sharing from [21].

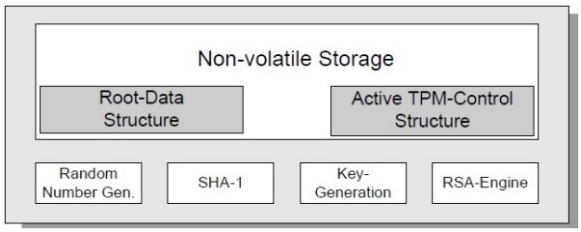


Figure 7.2: Layout of the multi-context TPM from [85].

authors do not elaborate on TPM migration and the management of TPM endorsement credentials in their paper.

7.3.2 Enhancing TPM with Hardware-Based Virtualization Techniques

The paper [85] presented the design of a TPM that supports hardware-based virtualization. In this design, the VMM time multiplexes the hardware TPM in a manner similar to [21]. However, a difference is that the hardware TPM has to be modified to include additional non-volatile memory to store the various TPM contexts. The layout of this multi-context TPM is shown in Figure 7.2.

The VMM manages the transition of one TPM context to another using the TPM control structure. The VMM links every TPM context to its own TPM control structure. Figure 7.3 shows the content of the TPM control structure. When a new TPM context is to be loaded, the VMM will store the previous TPM control structure and

Fields of the TPM Control Structure
Storage Root Key (SRK)
PCRs [16..23]
Attestation Identity Keys (AIK)
Endorsement Key (EK)
Endorsement Credential
Monotonic Counter Values
Values of the non-volatile storage
Delegation Tables
TPM context data
DAA TPM specific secret (f)
TPM_PERMANENT_FLAGS/DATA
TPM_STCLEAR_FLAGS/DATA
Authorization data

Figure 7.3: TPM control structure from [85].

load the new TPM control structure. To protect the confidentiality of the TPM control structure, it is encrypted and the cryptographic key is stored in the TPM root data structure.

Another feature is the concept of protection rings in the TPM. This TPM protection ring has a two level hierarchy that differentiates a privileged TPM mode from a non-privileged TPM mode. The TPM protection ring leverages the Intel VT architecture and hence can be considered as a form of hardware-based protection. There are two forms of CPU operation in the Intel VT architecture. The VMX root operation in which the VMM runs and the VMX non-root operation in which the VM runs. Only the VMX root can run the privileged TPM mode while the VMX non-root can only run in the non-privileged TPM mode. In addition, the authors extended the TCG specification for TPM to include extra commands to manage the transition between TPM modes and to control the different TPM contexts. These extra TPM commands can only be executed by VMX root.

For VM migration, the authors described that their TPM context migration protocol is similar to the concept TCG introduced for migratable keys. On TPM credentials, the authors proposed to establish a certificate chain with the root Endorsement Key (EK). For every TPM context, the TPM generates a new EK which is then certified by the root EK. When the TPM context is migrated, the EK will then be re-certified with the root EK of the destination platform.

This design covers many technical aspects of para-virtualizing TPMs, including credentials and migration. However, the TPM 2.0 is a new design and we shall examine in the following section if it is suitable for para-virtualization.

7.4 Examining TPM 2.0 Suitability for Para-Virtualizing

TPM 2.0 core functions are found to be generally suitable for use in a para-virtualized design and only some typical virtualization functions are required at the VMM

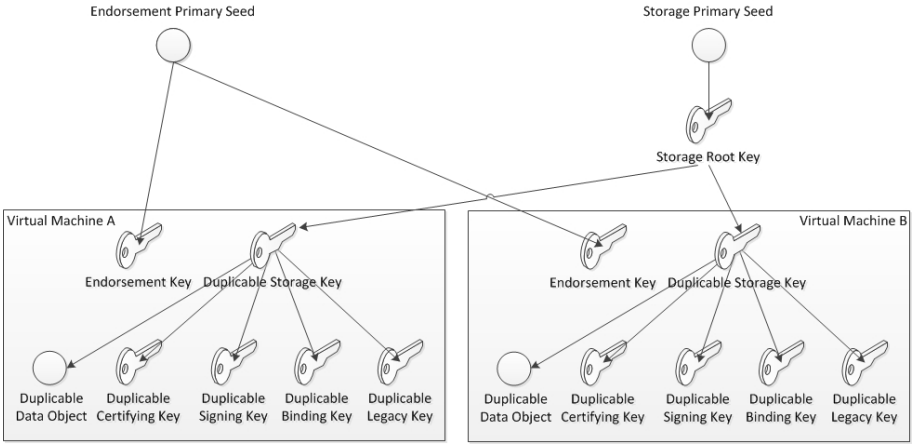


Figure 7.4: TPM 2.0 key distribution for multiple VM .

level to allow a single TPM 2.0 device to support multiple VMs. Our analysis of the suitability of TPM 2.0 core functions for para-virtualized are described in the following paragraphs.

7.4.1 Endorsement and Storage Keys

Figure 7.4 shows how the TPM keys can be distributed to multiple VMs. An endorsement key is derived from the endorsement primary seed located inside the TPM and it is the basis for the root of trust of reporting. Several endorsement keys can be generated and assigned to the various VMs hosted on the computing platform. Thus, an endorsement key can be migrated together with the associated VM. However, we recommend to obtain a new endorsement key after VM migration from the destination TPM since the host computing platform has changed. Consequently, a new certificate will have to be obtained for the new endorsement key after VM migration.

For keys derived from the storage root key, they can be created to be migratable by setting the duplication flag. These keys are assigned to the VMs and kept outside the TPM. As an example, when a VM wishes to use the TPM for certification, the certifying key is loaded into the TPM using the command TPM2_Load. At the end of the session, the certifying key is unloaded from the TPM. The TPM can then start a new session with another VM. During VM migration, a certifying key can be packaged as a duplicable data object using TPM2_Duplicate and then migrated over to the designated TPM. TPM2_Import will load the migrated data object into the destination TPM. Meanwhile, the use of a TPM authorization session will ensure that only the right VM can access its certifying keys. The resulting effect is comparable to the notion of process and resource isolation between VMs. Nevertheless, the credential for a certifying key may have to be re-issued after a migration as the host computer has changed.

7.4.2 Protected Storage

There are various TPM 2.0 commands that can move data in and out of either the volatile or non-volatile memory: for example, TPM2_Load, TPM2_LoadExternal, TPM2_Unseal, TPM2_NV_Write and TPM2_NV_Read. The use of TPM authorization data will ensure that only the right VM can access its data in the protected storage. The mechanism for the migration of these data in protected storage is the same as those for certifying keys. For data that are encrypted and where the access control depends on the host computing platform integrity measurements, this can cause a problem during VM migration when the host computer platform is different. The use of TPM 2.0 Enhanced Authorization will allow a more flexible access control policy for this type of protected data. For example, the authorization policy can either check the host platform integrity measurements or other security properties. In the meantime, the amount of volatile and non-volatile memory has to be managed to avoid a situation whereby there is insufficient memory for use by all the VMs on that host computer.

7.4.3 Integrity Measurement and Reporting

TPM2_NV_DefineSpace can be used to create PCR banks in the NV memory. TPM2_PCR_Extend will then be used to record the integrity measurements of the VMs into the PCRs located in the NV memory. For the host machine, TPM2_PCR_Extend will be used to record the integrity measurements into the PCRs located in the volatile memory. This arrangement will allow the integrity measurement of both the virtual and host machine be stored inside the TPM at the same time. As above, the amount of NV memory has to be managed to avoid the situation whereby there are more VMs than the TPM can support. TPM2_Quote is used to report the integrity measurement stored in a particular PCR. When reporting the integrity measurements to a requestor, TPM2_Load is used to insert the relevant attestation key into the TPM. This key is then used to sign the integrity measurement. The mechanisms for the migration and access control of PCRs are the same as those for certifying keys.

In addition to the points above, TPM 2.0 is designed with a context management feature that is intended to be used to manage TPM resources among various applications. In a virtualized environment, this feature can instead be used to manage TPM resources among various VMs. The TPM 2.0 specification states that the structure of the context is decided by the vendor. In other words, there can be a customized context structure to support TPM 2.0 para-virtualization requirements.

As with most hardware virtualization, TPM 2.0 will require some software components at the VMM level to allow it to support multiple VMs. For example, a software component is required to provide some form of usage management to ensure that every VM has fair use of the TPM. Another software component is required to block state altering TPM commands issued by non-management VMs. The architectures described in the TCG's virtualized trusted platform specification [88] are more suited to the full virtualization technique although certain aspects such as TPM migration are applicable to this para-virtualized TPM framework.

7.5 Requirements for Para-Virtualizing TPM 2.0

As discussed in Section 7.4, TPM 2.0 core functions are generally able to support para-virtualization. However, we noted that there have to be mechanisms to assign and ensure the fair use of TPM resources to multiple VMs. In addition, issues pertaining to an Enterprise IT environment, such as migration, certification and logging have to be addressed as well. VM migration occurs when a VM is moved from one physical computer to another. This process is commonly carried out if the VM requires a differently specified physical computer to support its computation tasks. VM migration can also happen when the owner moved the VM from one physical computer to another to facilitate maintenance activities. To this end, the design requirements presented in [77], [64], [21] and [85] were considered and we suggest the following:

1. The way that an application uses the para-virtualized TPM should be the same as for a hardware TPM.

2. The para-virtualized TPM should always be available for use by the VM.

3. Strong association between the VM and its TPM resources. This association should be maintained after the migration of the VM.

4. TPM resources belonging to a VM should not be accessible by another VM.

5. The size of both volatile and non-volatile memory in the TPM should support the additional memory required to host multiple VMs on a single physical computer.

6. Data stored in protected storage locations should be preserved unless instructed by their owners. These data should be moved together with the VM during migration and then stored into the protected storage location of the destination TPM.

7. The security strength of protected storage location in a para-virtualized TPM should be the same as for a hardware TPM.

8. The activity of the para-virtualized TPM should be logged. The log file associated with a particular VM should be moved to the destination host computing platform during VM migration.

9. Non-privileged VMs cannot execute commands that can alter the state of the TPM.

10. The para-virtualized TPM should have verifiable credentials at all times.

11. Individual VM interaction with the para-virtualized TPM should be isolated from each other to avoid interference.

12. The keys of a para-virtualized TPM should be backed up as part of business continuity planing.

7.6 An Enterprise Framework for Para-Virtualizing TPM 2.0

The framework shown in Figure 7.5 contains components at the VMM and hardware level to support the para-virtualization of TPM 2.0. This framework allows the multiplexing of TPM 2.0 functions and resources for use by VMs and their applications at the same time while preserving the hardware based protected storage. The development of this framework is based on the survey of TPM para-

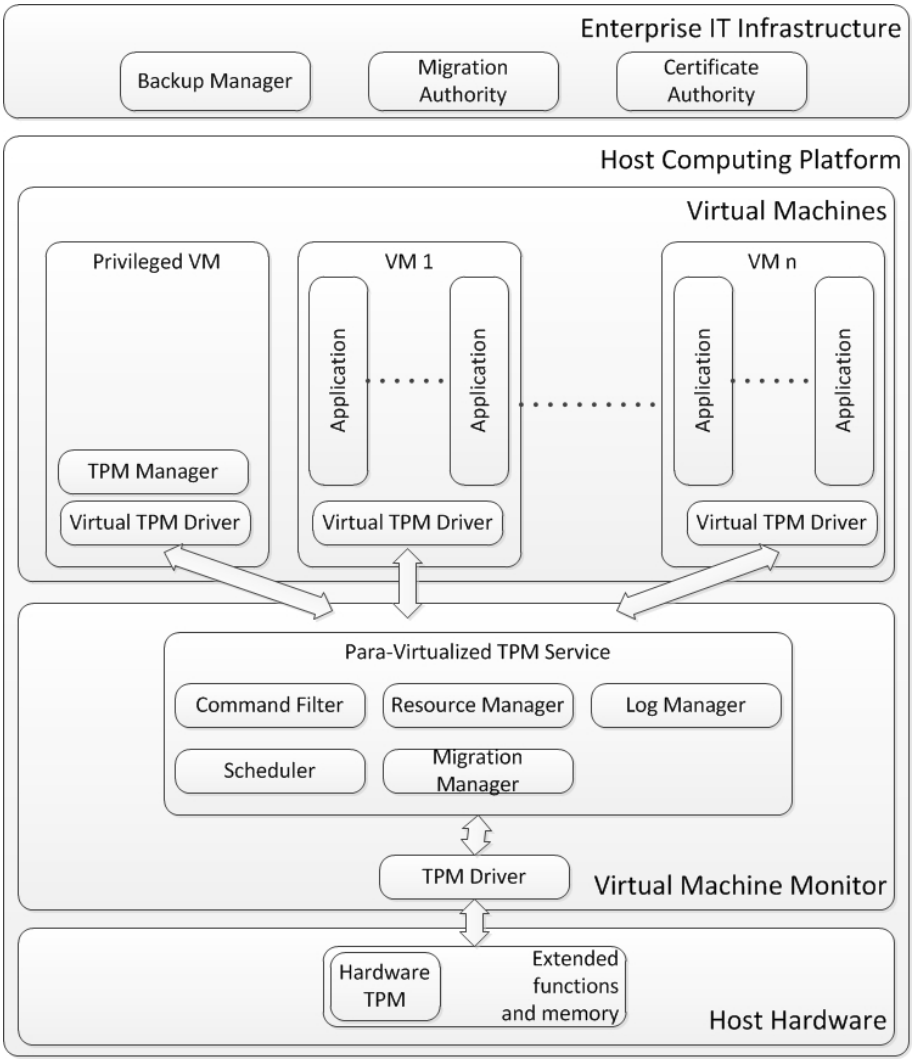


Figure 7.5: Enterprise framework for para-virtualizing TPM 2.0.

virtualizing works in Section 7.3, the analysis of para-virtualizing TPM 2.0 core functions in Section 7.4 and the design requirements from Section 7.5 of this chapter.

As shown in Section 7.4, the TPM 2.0 core functions are generally suitable for use in a para-virtualized design. Hence, the software components at the VMM level do not emulate TPM functions but instead focus on ensuring fair usage of TPM and on addressing requirements such as migration and logging.

A major difference from existing concepts is the removal of the virtual PCRs as the store for integrity measurements of VMs because they will be stored in the TPM NV memory. As a result, VM integrity measurements enjoy the security of hardware based protected storage. In addition, a privileged VM is used to manage the hardware TPM and para-virtualized TPM service. This gives the system administrator a separate conduit to control the hardware TPM and para-virtualized TPM service. There are also provisions for TPM hardware enhancements and a log

manager. These are security features desired by a high security Enterprise IT infrastructure. Moreover, this framework covers external components that are essential for the proper functioning of the para-virtualized TPM in an Enterprise IT environment. The following subsections describe the components in this framework.

7.6.1 Extended functions and additional memory

The framework allows modifications to be carried out on the TPM hardware to support the requirements of the virtualized environment. For example, the amount of memory can be increased to include more PCR banks to store the integrity measurements of multiple VMs. Although hardware modifications can be costly, certain high security requirements, for example, government Enterprise IT, may demand and pay for this enhancement. In addition, the modified hardware can contain extended functions to support new virtualization techniques such as the single root I/O virtualization standard specified by the PCI-SIG [62]. Hardware techniques such as the use of field-programmable gate arrays described in [65] can be used to implement the modifications. Depending on security requirements and the amount of modification, a new platform certificate may have to be issued for the modified TPM to vouch that the platform contains a genuine TPM and that the communication path between the TPM and the host computer is trusted.

7.6.2 Command filter

As there are commands that can alter the TPM state, for example TPM2_Shutdown, it is necessary that only the TPM manager in the privileged VM can execute such commands. All commands from the VM to the hardware TPM will be inspected and any state-altering command from a non-privileged VM will be blocked from execution. The command filter can make use of the hardware based virtualization technique described in [85] to enforce the restriction on the use of selected TPM commands.

7.6.3 Scheduler

This component sequences VM access to the hardware TPM in a time division multiplexing manner. This will ensure that every VM has an opportunity to use the TPM. The algorithm for sequencing can be either round robin or demand based. In round robin, the scheduler can poll every VM and ask if it likes to use the TPM. However, it is a challenge to decide the optimal time allocated for each request. If the time is set too short, then too much switching from one VM to another will take place and the round robin design will become slow. If the time is set too long, then time could be wasted if a VM uses the TPM for less than the allocated time. Alternatively, the scheduler can arrange access to the TPM whenever the VM makes a request. The time that a VM can use the TPM will be limited. If the TPM is currently servicing a request, any new request will be queued in a first-in-first-out memory cache.

7.6.4 Resource manager

To achieve strong association between VMs and their TPM resources, the resource manager will administer the assignment of TPM resources such as keys. It is envisioned that the VM will use keys derived from the primary seed in the hardware TPM. In turn, each VM will build their key hierarchy based on their assigned key. Meanwhile, the resource manager will create PCR banks in the TPM NV memory and assign them to the VM. PCRs in the TPM volatile memory will be reserved for use by the host computer and VMM. For the purpose of attestation, the resource manager can provide the PCR contents in the TPM volatile memory to the VM. The other task carried out by the resource manager is to work together with the scheduler to isolate the TPM processes and resources of the VM. The mechanisms used to achieve this effect can include the use of the TPM context management feature and authorization sessions. TPM commands such as TPM2_ContextSave, TPM2_ContextLoad, TPM2_FlushContext and TPM2_StartAuthSession will be employed. This will prevent one VM from accessing the TPM resources of another VM. TPM contexts are saved in the VMM.

7.6.5 Migration manager

During VM migration, this component will work together with the resource manager to oversee the packing of the associated TPM resources into duplicable data objects. The command TPM2_duplicate will be used to prepare the data object for migration. This component will also work with the migration authority to operate a migration protocol that securely moves the duplicated data object to the designated TPM. On the other hand, if the host computer is to receive a migrated VM, the migration manager can carry out the task of authenticating and verifying the trustworthiness of the migrating VM.

7.6.6 Log manager

The availability of a log is crucial for forensic investigation in the event of a security incident. The log manager will log all the operations performed by the VM on the hardware TPM. The log manager can make frequent integrity measurements of the log file and store the measurements in the hardware TPM. This will allow the detection of unauthorized changes to the log file. Meanwhile, the log manager with work together with the migration manager and migration authority to move the log file associated with a particular VM to the destination host computing platform during VM migration.

7.6.7 TPM manager

This resides in a privileged VM and is primarily used to manage the hardware TPM and configure the para-virtualized TPM service. The privilege status of this VM can either be enforced from the VMM or controlled by the hardware based virtualization technique described in [85]. The TPM manager can check the integrity of the VMM components by querying the hardware TPM.

7.6.8 TPM and Virtual TPM driver

The TPM driver contains the software stack that enables the VMM to communicate with the hardware TPM while the Virtual TPM driver contains the software stack that enables the VM to communicate with the para-virtualized TPM service in the VMM. To detect unauthorized changes to these two components, integrity measurements are carried out when they are started. The integrity measurements are then stored in the TPM.

7.6.9 Backup manager

This and the next two components are external to the computer platform but part of the Enterprise IT infrastructure. To support business continuity planning, the TPM keys for the VM should be archived and stored securely in a physically separate location. In the event of an incident that causes the TPM keys to be lost, the backup TPM keys can be retrieved to support the continuation of business operation.

7.6.10 Migration authority

Besides administrating the migration of VMs, the migration authority can work with the TPM migration manager to ensure that the accompanying TPM resources are moved to the correct destination TPM.

7.6.11 Certificate authority

This component will verify the credentials of TPM keys provided by the VM and issue the appropriate certificates when it is satisfied with the credentials. After a VM has been migrated, it can work with the migration authority to recheck the credentials of the TPM keys and issue new certificates. More importantly, the certificate authority can revoke a particular certificate when required.

7.7 Requirements Revisited

To assess the thoroughness of the design of this framework, it is compared to the requirements listed in Section 7.5. Firstly, there are no changes to the TPM commands and most TPM commands are available to the VM except for those that can alter the TPM state. This meets the requirement of retaining the same TPM usage model. Secondly, the scheduler will sequence the TPM commands issued by the VMs. Besides ensuring that every VM can interact with the TPM, it works in conjunction with the resource manager to switch from one VM's TPM session to another session. The resource manager will link the TPM resource to the VM and access control is achieved by using authorization sessions. The resulting effect is equivalent to isolating each VM's interaction with the TPM.

Meanwhile, the migration manager works with the resource manager to maintain the association between a TPM resource and its VM during migration. In addition, this framework allows for modifying the TPM hardware and this provision gives the flexibility to complement TPM hardware with additional memory to meet the TPM memory requirement of multiple VMs. TPM 2.0 NV memory is

a protected data storage of this framework. The resource manager can store a certain number of integrity measurements from numerous VMs into this hardware memory. Hence, the security protection of integrity measurement in this para-virtualized TPM is the same as for a plain hardware TPM. The other requirement relating to logging is fulfilled by the log manager while the backup manager is used as part of the business continuity plan. As for certification, this framework uses an external certificate authority to check on the credentials of TPM keys. When it is satisfied with the credentials, it will issue a certificate to vouch for the TPM keys. Last of all, the command filter makes sure that non-privileged VM cannot execute commands that can alter the TPM state. To conclude, this framework meets the requirements set forth in Section 7.5.

7.8 Summary

We found that TPM 2.0 core functions are generally suitable for para-virtualization. This indicates that the technical barrier to using TPM 2.0 in a virtualization environment can be potentially lowered. The proposed framework is holistic as it covers important considerations at different levels of the virtualization environment. Differences from existing concepts include storing integrity measurements of VMs in TPM NV memory and using a privileged VM to manage the hardware TPM and para-virtualized TPM service. There are also provisions for TPM hardware enhancements and a log manager. Moreover, this framework covers external components that are essential for the proper functioning of the para-virtualized TPM in an Enterprise IT environment. Thus, the studies and framework expressed in this chapter provide a comprehensive basis for future work in para-virtualizing TPM 2.0 and integrating the design with an Enterprise IT virtualization environment.

Chapter 8

Conclusion and Future Work

8.1 Thesis Summary

At the beginning of this thesis, we provided some questions to guide this research on identifying trust properties. The first part of this thesis describes how we answer these research questions. To address the understanding of trust by computer users, we conducted a socio-technical study on trust notions and obtained a list of concepts that computer users associated with trust in a computing device. More importantly, this list affirms the properties of trust in this thesis. Then we proceeded to work on Grawrock's challenge of a language that could describe the trustworthiness of a computing device. In this aspect, we researched and developed a novel causality-based model to describe the trustworthiness of a computing device. This model provides information on the causal dependencies between trust notions, capabilities, mechanisms and configurations. We made use of both semantic and set theoretic definitions to specify this model. Thereafter, we implemented the model in the TNC architecture. In addition, we discussed how this description can be evaluated. In the next step, we worked on using attestation to convey assurance on the trustworthiness of a computing device. We made a point about the limitations of binary attestation and set about introducing provenance-based attestation as an alternative. We presented a design and leveraged the PROV data model for this method of attestation. We examined approaches to implementation and looked at how a provenance record can be evaluated. Then we analysed the design using threat modelling and produced an attestation protocol. Consequently, we gained insights into the properties of trust and how they can be expressed, evaluated and attested.

The second part of this thesis describes how we answer the research questions we provided on developing trusted systems. To address the sharing of information on trusted hardware between experts and developers, we crafted an ontology of a computing device secured with TPM 2.0. This ontology was recorded in XML format and we demonstrated how a developer can obtain information from the ontology by using SPAQRL queries. On the method to identify threats to use scenarios of trusted hardware, we developed a use scenario of TPM 2.0 and studied the use of threat modelling to identify threats and suggest mitigations. Subsequently, we worked on an application of trusted hardware. We examined how TPM 2.0 can be used in a modern system by putting forward a framework for para-virtualizing TPM 2.0. This framework will be of use to those who wish to use the newer TPM in the Enterprise IT virtualization environment. As a result, we have gained information on using TPM 2.0 as a building block for trust properties in trusted systems.

8.2 Future Work

In the following sub sections, we discuss the plethora of research still to be undertaken to address limitations in our work.

8.2.1 Study on Trust Notions

A limitation of this study is that the survey sample size could be larger to better represent the population. In addition, more in depth studies, for example face to face interviews, could be conducted to better examine the conceptual relationship of trust, stake and risk proposed by Solhauf et al [82]. Finally, the study could be made more focused by reducing the number of notions.

8.2.2 Causality-based Model

Our causality-based model at this stage lays the foundation for two types of future work. The first type refers to the concept of predictions in causal models. For example, an evaluator can predict the trustworthiness of a computing device if it is presented with causal graphs describing the trust notions and capabilities. This type of future work will be a progression from our original intent of developing the causality-based model for end-to-end trust. The second type refers to the concept of interventions in causal models. For example, a security engineer can examine reference causal graphs and adjust either the mechanisms or configurations of a computing device to make sure that certain trust notions and capabilities are achieved. These two types of future work will require the development of algorithms that could understand the semantics of the causal graph and subsequently carry out intelligent processing.

We mentioned in Section 3.2 that our causality-based model does not have the ability to deal with the probability of a causal dependency. However, if we expand this causality-based model to include the probability of a causal dependency, then we can develop sophisticated models that predict the trustworthiness of a computing device in an uncertain environment. For example, we can model a weak causal dependency to reflect an attack and find out how the trust notions and capabilities are affected. This ability can be used in a cyber test range where it requires the modelling of computing devices under attack [8], [98]. This future work will require the use of Bayesian networks that employ probabilistic and statistical models.

8.2.3 Provenance-based Attestation

The provenance-based attestation described in Chapter 4 is feasible but will require more work before it can be considered as mature. A key area for improvement is establishing the trustworthiness of provenance itself. We mentioned in Table 4.1 the use of digital signatures to mitigate the threat of an attacker creating a false provenance record. This mitigation can be further strengthened when we consider other works on this issue. For example, Hartig proposed extending RDF to include trust information and the use of a modified SPARQL for trust assessment [31]. Meanwhile, Groth et al. described a protocol for recording the documenta-

tion of a system execution [26]. This protocol ensures that the documentation can accurately record provenance data.

The other limitation is the lack of a systematic methodology for capturing provenance data. Besides the techniques discussed in Section 4.6, we believe a robust way to capture provenance data is for all components of the software stack to log provenance data using a standard format such as the PROV data model. This will involve the developers of these components to modify program code. It will not be a trivial effort but the benefit can outweigh the inconvenience.

8.2.4 Ontology Development

A drawback of our ontology is that it only focus on TPM 2.0 and does not provide more information about other technologies that can interact with it. Thus, the ontology can be expanded or merged with other ontologies to provide more details of how other technologies can work with TPM 2.0. Besides technology oriented ontologies, our ontology can also be merged with ontologies that describe abstract concepts of cyber security [58]. This is because our ontology features the mapping of low level technical primitives to high level trust notions. Finally, ontology development is a continual process. Classes and properties should be added or deleted to address new developments.

8.2.5 Threat Modelling

The threat model has not been validated to ensure the quality and accuracy of the threats and mitigations. This involves checking that all the potential threats are identified and the mitigations are assessed to be adequate. Shostack has written about guidelines to validate the threat models [79]. However, this is still a manual process and we are not aware of any fully automated threat modelling process.

8.2.6 Para-Virtualizing TPM 2.0

This framework is based on a paper study of the TPM 2.0 specification. From the view point of a high level design, this framework can be considered to be reasonable. To validate this framework, the components in the VM monitor have to be implemented and tested. An ideal candidate for the VM monitor is the open source Xen hypervisor [13]. The Xen hypervisor can support para-virtualization and hardware based virtualization technology such as Intel VT and AMD V CPU architecture extension. The hypervisor has a driver for the TPM although it is for the TPM 1.2 specification.

The techniques described in [21], [75] and [85] can be redeployed and used to implement this framework but significant challenges still exist. The performance of this para-virtualized TPM design is one area to be investigated further. Research can be carried out to develop better algorithms in the scheduler to sequence VM requests for TPM resources. The strength of the isolation between each VM interaction with the para-virtualized TPM is another research area to be studied. The commonly used integrity measurement method of obtaining a digest from hashing a software component can be difficult to implement in a virtualized environment.

This is because VM migration can take place and the configuration of host computing platforms can differ. Attestation in such an environment will be tedious as there will be a variety of measurements to contend with. Hence, the provenance-based attestation presented in Chapter 4 of this thesis can be examined for use in this environment as it does not rely on integrity measurements.

The development of use scenarios for this framework is a task not to be neglected. It will be difficult to persuade organizations to take up this framework if there are no functional use scenarios. Meanwhile, the threat modeling process studied in Chapter 6 can be conducted on these scenarios. The results can be used by researchers to harden the design, and organizations who wish to adopt the framework can put in mitigation measures recommended from the threat model to avoid any pitfalls. Lastly, the use of the TPM monotonic counter by VMs is not addressed in the proposed framework and further work can be done to study this matter.

8.3 Conclusion

The issue of identifying trust properties is particularly challenging due to the varied understanding of trust and the difficulty in linking these understandings to the underlying trust primitives. The work presented in this thesis addresses these challenges by bringing together the two distinct research domain of trust notions and trust primitives. We consider the causality-based model as our main contribution as it could describe, in a novel way, the trustworthiness of a computing device with reference to trust notions, capabilities, computing mechanisms and their configurations. This contribution is made comprehensive as we have conducted a study on trust notions and proposed using provenance data as trust evidence to attest to components in the trust description. Thus, we are now in a position where we can begin to further explore the trust language envisioned by Grawrock. In addition, we have gained more information on using TPM 2.0 as a building block of trust properties through our work on ontology, use scenario, threat modelling and paravirtualization. These contributions extend our knowledge on developing trusted systems.

Appendix A

A Socio-Technical Study on User Centered Trust Notions and Their Correlation to Stake in Practical Information Technology Scenarios - Secondary Analyses

The online survey collected a wealth of data on how different demographic groups rank the subjective and objective trust notions in the two scenarios. Although the analyses of these data do not contribute towards answering the research questions, we thought we should present the findings in this thesis for those who are interested.

A.0.1 Scenario One

A number of other hypothesis tests using two tailed independent sample t-test at 5% significance level were carried out. These hypothesis tests aimed to analyse the results further to find out if there was any statistical difference in the mean rank of notions among different segments of the demographics. Each hypothesis test was carried out once for the six subjective trust notions and eight objective trust notions. The results of the hypothesis tests are described in the following.

Hypothesis testing 3

H0 : The mean rank of the subjective and objective trust notions between male and female are equal. H1 : The mean rank of the subjective and objective trust notions between male and female are not equal.

Analysis. There are 67 female respondents and 88 male respondents. H0 is rejected for the objective notion of security standard. The male respondents are more concerned about security standard (mean rank of 4.22 against 4.90) than female respondents.

Hypothesis testing 4

H0 : The mean rank of the subjective and objective trust notions between respondents of bachelor degree or higher and those of associate degree or high school graduates are equal. H1 : The mean rank of the subjective and objective trust notions between respondents of bachelor degree or higher and those of associate degree or high school graduates are not equal.

Analysis. There were 106 respondents with bachelor degree or higher qualification and 50 respondents with associate degree or high school qualification. The hypothesis test was carried out for each of the six subjective trust notions and each of

the eight objective trust notions. H0 is rejected for the subjective notion of aesthetics. Respondents with bachelor degree or higher place less emphasis on aesthetics (mean rank of 5.20 against 4.56) compared to respondents with associate degree or high school qualification. In addition, H0 is rejected for the objective notion of identity and authenticity of the mass market laptop computer, data confidentiality and the availability of required function or data. When compared with respondents with associate degree or high school qualification, respondents with bachelor degree or higher consider data confidentiality (mean rank of 2.33 against 3.38) to be more important while giving less importance to identity and authenticity of the mass market laptop computer (mean rank of 4.86 against 4.30) and the availability of required functions or data (mean rank of 5.31 against 4.78).

Hypothesis test 5

H0 : The mean rank of the subjective and objective trust notions between respondents who are early adopters and those who are minimalist are equal. H1 : The mean rank of the subjective and objective trust notions between respondents who are early adopters and those who are minimalist are not equal.

Analysis. There were 86 respondents who were early adopters and 60 respondents who were minimalists. The hypothesis test was carried out for each of the six subjective trust notions and each of the eight objective trust notions. H0 is rejected for the objective notion of identity and authenticity of the mass market laptop computer and availability of required functions or data. The results show that early adopters give less importance availability of required functions or data (mean rank of 5.43 against 4.87) but more importance to identity and authenticity of the mass market laptop computer (mean rank of 4.36 against 5.07). This could be because early adopters understand that new technologies may still require some improvement to work perfectly and it is alright that required functions or data are unavailable occasionally. However, early adopters are concerned that this new technology must come from the original manufacturer and not be a fake.

A.0.2 Scenario Two

A number of other hypothesis tests using two tailed independent sample t-test at 5% significance level were carried out. These hypothesis tests aimed to find out if there was any statistical difference in the mean rank of notions among different segments of the demographics. Each hypothesis test was carried out once for the six subjective trust notions and eight objective trust notions. The results are described in the following.

Hypothesis test 6

H0 : The mean rank of the subjective and objective trust notions between male and female are equal. H1 : The mean rank of the subjective and objective trust notions between male and female are not equal.

Analysis. There were 74 female respondents and 81 male respondents. The hypothesis test was carried out for each of the six subjective trust notions and each of the eight objective trust notions. H0 is not rejected in all the tests. This outcome is different from the same hypothesis test result for scenario one.

Hypothesis test 7

H0 : The mean rank of the subjective and objective trust notions between respondents of bachelor degree or higher and those of associate degree or high school graduates are equal. H1 : The mean rank of the subjective and objective trust notions between respondents of bachelor degree or higher and those of associate degree or high school graduates are not equal.

Analysis. There were 105 respondents with bachelor degree or higher qualification and 50 respondents with associate degree or high school qualification. The hypothesis test was carried out for each of the six subjective trust notions and each of the eight objective trust notions. H0 is rejected for the subjective notion of usefulness, aesthetics, vendors technical competence and vendors interest in respondents wellbeing. The results show that respondents holding bachelor degree or higher place more emphasis on the usefulness (mean rank of 3.01 against 3.92) and less emphasis on aesthetics (mean rank of 5.13 against 4.72), vendors technical competence (mean rank of 3.38 against 2.96) and vendors interest in respondents wellbeing (mean rank of 3.96 against 3.54). This observation for the subjective notion of aesthetics is the only similarity with the observations from hypothesis testing of the same demographic segment in scenario one.

Hypothesis test 8

H0 : The mean rank of the subjective and objective trust notions between respondents who are early adopters and those who are minimalist are equal. H1 : The mean rank of the subjective and objective trust notions between respondents who are early adopters and those who are minimalist are not equal.

Analysis. There were 76 respondents who are early adopters and 72 respondents who are minimalists. The hypothesis test was carried out for each of the six subjective trust notions and each of the eight objective trust notions. H0 is rejected for the objective notion of identity and authenticity of the embedded computing devices. Early adopters consider this notion (mean rank of 4.38 against 5.04) to be more important and this is the same observation for early adopters in scenario one.

Appendix B

Threat Modelling Report of the Provenance-based Attestation Design

Threat Model: User application use TPM to encrypt data for group share

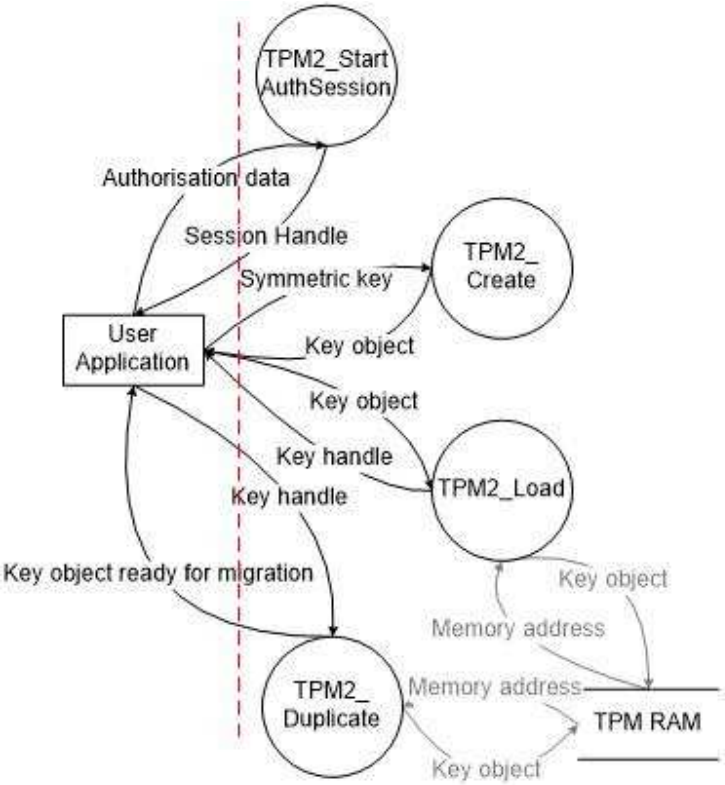
Threat Model Information
Data Flow Diagrams
Threats and Mitigations
Certifications
External Dependencies
Implementation Assumptions
External Security Notes

Threat Model Information

Component	User application use TPM to encrypt data for group share
Product	Trusted Platform Module 2.0
SourceUrl	
Owner	Trusted Computing Group
Participants	Jiun Yi Yap, Allan Tomlinson
Reviewers	
Url	
Summary	A PhD research project at Royal Holloway University of London Information Security Group
History	

Data Flow Diagrams

Context



Threats and Mitigations

Elements

Element Type	Description
Data Flow	Authorisation data
Data Flow	Key handle
Data Flow	Key handle
Data Flow	Key object ready for migration
Data Flow (NotGenerated)	Key object (this element is not accessible from entry points)
Data Flow (NotGenerated)	Key object (this element is not accessible from the entry points)
Data Flow	Key object
Data Flow	Key object
Data Flow (NotGenerated)	Memory address (This element is not accessible from the entry points.)

Data Flow (NotGenerated)	Memory address (This element is not accessible from the entry points.)
Data Flow	Session Handle
Data Flow	Symmetric key
Data Store	TPM RAM
Interactor	User Application
Process	TPM2_ Create
Process	TPM2_ Duplicate
Process	TPM2_Load
Process	TPM2_StartAuthSession
TrustBoundary	

External Interactors

Threats against User Application

Spoofing (Threat #25)

Threat: Attacker uses social engineering methods to gains access to the computing system and attempt to interact with TPM.

Mitigation: This threat can be mitigated if the use of TPM is combined with more sophisticated authentication methods, eg smart card, 2FA.

Most TPM commands require authorisation. TPM authorised a user based on its knowledge of a secret credential. If the authorisation fails, the user will not be able to carry out the commands. However, the credential has to be stored securely in the TCB.

Spoofing (Threat #121)

Threat: Attacker eavesdrops on authorisation session between user application and TPM.

Mitigation: The salt used to generate the session key is encrypted by a key protected by TPM. The session key in turns encrypt the data exchanges during the session between the user application and TPM.

Spoofing (Threat #144)

Threat: Attacker inserts malicious codes into the computing system and attempt to interact with TPM

Mitigation: TPM will have to reply on the security of the TCB to stop malicious codes from entering the computing system. The integrity measurement at system start up can protect against this type of threat but there is a risk of run time attacks.

Most TPM commands require authorisation. TPM authorised a user based on its knowledge of a secret credential. If the authorisation fails, the user will not be able to carry out the commands. However, the credential has to be stored securely in the TCB.

Spoofing (Threat #145)

Threat: Attacker makes changes to the authorisation value in TPM either by gaining unauthorised access to the computing system or by inserting malicious codes.

Mitigation: Changes to the authorisation value in TPM will require the authorisation of the administrator.

However, the administrator credential has to be stored securely in the TCB.

Spoofing (Threat #146)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Repudiation (Threat #26)

Threat: User denies carrying out certain commands.

Mitigation: TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #124)

Threat: Attacker pretends to be a user application and replays commands.

Mitigation: TPM sessions make use of nouces to mitigate against replay attacks.

Repudiation (Threat #125)

Threat: Attacker accesses the computing system and edits log to misrepresent the record of commands.

Mitigation: Besides strong authenication of user, TCB has to make sure only authorisation changes can be made to the log.

Processes

Threats against TPM2_ Create

Spoofing (Threat #130)

Threat: Attacker creates a data object outside the TPM and attempt to pass off the object as one that is created by TPM.

Mitigation: TPM can attest to the data that it has created by using the command TPM2_Hash and TPM2_Sign. TPM2_Hash will validate the object to make sure that it is indeed created by the TPM and output a digest if successful. TPM2_Sign will then attest to the object by producing a digital signature. The parent key used for digital signature can be installed during manufacturing and certified by the manufacturer for authenticity.

Spoofing (Threat #133)

Threat: Attacker pretends to be user application and attempt to execute this command.

Mitigation: The integrity measurement at system start up can detect the presence of unauthorised software. However, there is a risk of executing unauthorised software during run time and the security features of the TCB will have to mitigate this threat.

This command is only authorised to be used by the user role and the command will check for successful authorisation before executing. However, the credential has to be stored securely in the TCB. In addition, the TCB will have to authenticate the user.

Spoofing (Threat #134)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Tampering (Threat #86)

Threat: Attacker carries out a buffer overflow attack in an attempt to access other data stored in the TPM.

Mitigation: TPM2_Create validates the input data before executing the process of creating the new object. If input validation fails, the command will not execute. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of adnormal behaviour due to memory buffer overflow attack is minimal.

Tampering (Threat #147)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Tampering (Threat #148)

Threat: Attacker injects a compromised primary seed into the TPM and attempts to break the cryptographic protections that uses the compromised primary seed.

Mitigation: When the TPM is first manufactured, a primary seed can be injected into the TPM and the manufacturer will provide a credential to that. This process can be audited by a trusted third party. When the TPM is used, the credential can be checked to ensure authenticity.

The TPM primary seed can be changed after manufacture using the command TPM2_ChangeEPS. This command required physical presence authorisation. When a primary seed is changed, any existing key hierarchy will become void.

Repudiation (Threat #87)

Threat: User application denies that object is created by TPM.

Mitigation: The object produced by TPM2_Create will include an unique value (TPM_GENERATED_VALUE).

TPM will only allow a restricted signing key to sign an object that has this unique value. This unique value can only be produced by the TPM and it has integrity protection. As confidence in the attestation of an object is related to the confidence in the signing key, the highest confidence is provided by a restricted signing key that is created on the TPM with a certificate from the TPM manufacturer.

Repudiation (Threat #149)

Threat: User application denies executing this command on TPM.

Mitigation: TPM will have to rely on the TCB to keep a log of the commands performed on TPM. The availability of a log is crucial to forensic investigation in the event of a security incident. An example of a guideline for the security management of the log will be NIST SP 800-92.

Information Disclosure (Threat #131)

Threat: Attacker carries out memory buffer overflow attacks in an attempt to access data stored in TPM.

Mitigation: TPM2_Create carries out validation of input data. The command will fail if data validation is not successful. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of information disclosure due to memory buffer overflow attack is minimal.

Information Disclosure (Threat #140)

Threat: Attacker launches physical attacks on TPM to obtain parent seed value in an attempt to break the cryptographic protection for the created object.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #150)

Threat: Attacker attempts to weaken the cryptographic protection implemented by this command for the created object by modifying the cryptographic algorithm during firmware upgrade.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Information Disclosure (Threat #151)

Threat: Attacker injects a compromised primary seed into the TPM and attempts to break the cryptographic protections that uses the compromised primary seed.

Mitigation: When the TPM is first manufactured, a primary seed can be injected into the TPM and the manufacturer will provide a credential to that. This process can be audited by a trusted third party. When the TPM is used, the credential can be checked to ensure authenticity.

The TPM primary seed can be changed after manufacture using the command TPM2_ChangeEPS. This command required physical presence authorisation. When a primary seed is changed, any existing key hierarchy will become void.

Denial of Service (Threat #89)

Threat: Attacker attempts to use up all the TPM computing resources by sending this command repeatedly.

Mitigation: TPM relies on the TCB to mitigate this type of attack. The integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks. TPM does not have any feature to control resource consumption but there is dictionary attack protection for authorisation session.

Denial of Service (Threat #152)

Threat: Attacker attempts to use up all the TPM memory by providing malicious input paramenters.

Mitigation: TPM2_Create validates the input parameters before executing the process of creating the new object. If input validation fails, the command will not execute.

Elevation of Privilege (Threat #90)

Threat: Attacker attempts to elevate priviledge vertically by carrying out buffer overflow attacks through TPM2_Create.

Mitigation: TPM2_Create validates the input data before executing the process of creating the new object. If input validation fails, the command will not execute. In addition, priviledge commands will require another authorisation session.

Elevation of Privilege (Threat #153)

Threat: Attacker attempts to elevate priviledge horizontally by attempting to create a child object from a hierarchy that he does not have access to.

Mitigation: Authorisation is required for the use of parent object in this command.

Threats against TPM2_ Duplicate

Spoofing (Threat #97)

Threat: Attacker pretends to be user application and attempt to execute this command.

Mitigation: The integrity measurement at system start up can detect the presence of unauthorised software. However, there is a risk of executing unauthorised software during run time and the security features of the TCB will have to mitigate this threat.

This command is only authorised to be used by the duplication role. The command will check for successful authorisation and correct policy digest before executing. However, the credential has to be stored securely in the TCB. In addition, the TCB will have to authenticate the user.

Spoofing (Threat #160)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Tampering (Threat #98)

Threat: Attacker carries out a buffer overflow attack in an attempt to access other data stored in the TPM.

Mitigation: This command validates the input data before executing the process of duplicating the new object. If input validation fails, the command will not execute. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of adnormal behaviour due to memory buffer overflow attack is minimal.

Tampering (Threat #161)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Tampering (Threat #162)

Threat: Attacker edits an object and attempt to load it into TPM and duplicate it.

Mitigation: This command will check the integrity, attributes and cryptographic binding of the object before loading it into the TPM. The command will not execute and it return an error code if the check fails. If TPM2_Load executes successfully, an object handled will be returned to the user application which in turn pass

this handle to the TPM2_Duplicate command. TPM2_Duplicate command will check if this object is supposed to be duplicated and that the policy requirements are met.

Tampering (Threat #163)

Threat: Attacker attempts to import the duplicated object into another TPM that is not suppose to recieve this duplicated object.

Mitigation: The duplicated object can have two layers of encryption. The inner layer of encryption is carried out using a symmetric key generated by the source TPM. It is important to note that this symmetric key should meet security requirements.

The out layer of encryption is carried out using a symmetric key generated from a seed obtained from the destination TPM. It is importat to note that this seed should meet security requirements. This seed is shared between the source and destination TPM using asymmetric encryption. Hence, another TPM will not be able to import this duplicated object unless it has access to the seed.

Repudiation (Threat #99)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #164)

Threat: User application denies duplicating this object using the source TPM.

Mitigation: The outer layer of encryption is carried out using a key derived from a seed that is shared between the source and destination TPM using asymmetric encryption. However, the mechanism to share this seed must be secure.

As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Information Disclosure (Threat #100)

Threat: Attacker launches physical attacks on TPM to obtain seed value in an attempt to break the cryptographic protection for the duplicated object.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #165)

Threat: Attacker attempts to obtain the seed value used to provide the outer layer of encryption by cracking the sharing protocol which uses asymmetric encrpytion.

Mitigation: The public key infrastructure and cryptographic functions used for this purpose have to meet security requirements. The attestation feature of the TPM can also be used to provide some assurance on the authenticity and security of the endpoints.

Information Disclosure (Threat #166)

Threat: Attacker sends a compromised seed value to this command in an attempt to weaken the cryptographic protection for the duplicated object.

Mitigation: This command requires policy authorisation. The policy can include checks on the integrity of the computing system. This protects against the insertion of malicious codes that compromise the seed value sharing mechanism.

TPM has to rely on the TCB to check on the authenticity and integrity of the seed value. The destination TPM can send a digital certificate to vouch for the authenticity and integrity of the seed value. The source TPM can then verify the digital certificate with the Certificate Authority. Meanwhile, the detailed design and implementation for this mechanism has to meet security requirements.

Information Disclosure (Threat #167)

Threat: Attacker attempts to trick the source TPM to send the duplicated object to an unauthorised TPM.

Mitigation: The TPM has to rely on an external migration authority to securely manage the duplication of objects among TPMs. The migration authority should have features for authentication and authorisation and the implementation must meet security requirements.

Denial of Service (Threat #101)

Threat: Attacker attempts to use up all the TPM computing resources by sending this command repeatedly.

Mitigation: TPM relies on the TCB to mitigate this type of attack. The integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks. TPM does not have any feature to control resource consumption but there is dictionary attack protection for authorisation session.

Denial of Service (Threat #168)

Threat: Attacker launches DOS attack against the migration infrastructure.

Mitigation: TPM has to rely on the infrastructure provider to mitigate this type of attack. If all the computing systems in the network are equipped with TPM, then the integrity measurement and attestation function can provide a higher level of trustworthiness.

Elevation of Privilege (Threat #102)

Threat: Attacker attempts to elevate priviledge vertically by carrying out buffer overflow attacks.

Mitigation: Priviledge commands will require another authorisation session.

Elevation of Privilege (Threat #169)

Threat: Attacker attempts to elevate priviledge horizontally by attempting to duplicate an object that he does not have access to.

Mitigation: Authorisation is required for the use of object in this command.

Threats against TPM2_Load

Spoofing (Threat #91)

Threat: Attacker pretends to be user application and attempt to execute this command.

Mitigation: The integrity measurement at system start up can detect the presence of unauthorised software.

However, there is a risk of executing unauthorised software during run time and the security features of the TCB will have to mitigate this threat.

This command is only authorised to be used by the user role and the command will check for successful authorisation before executing. However, the credential has to be stored securely in the TCB. In addition, the TCB will have to authenticate the user.

Spoofing (Threat #154)

Threat: Attacker guesses the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #155)

Threat: Attacker creates a TPM object externally and attempts to load this object into TPM.

Mitigation: This command will check the integrity, attributes and cryptographic binding of the object before loading it into the TPM. The command will not execute and it return an error code if the check fails.

Tampering (Threat #92)

Threat: Attacker carries out a memory buffer overflow attack in an attempt to access other data stored in the TPM.

Mitigation: This command validates the input data before executing the process of creating the new object. If input validation fails, the command will not execute. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of adnormal behaviour due to memory buffer overflow attack is minimal.

Tampering (Threat #156)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Tampering (Threat #157)

Threat: Attacker edits an object and attempt to load it into TPM.

Mitigation: This command will check the integrity, attributes and cryptographic binding of the object before loading it into the TPM. The command will not execute and it return an error code if the check fails.

Repudiation (Threat #93)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #158)

Threat: User application denies loading this object to the TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Information Disclosure (Threat #94)

Threat: Attacker carries out memory buffer overflow attacks in an attempt to access data stored in TPM.

Mitigation: This command carries out validation of input data. The command will fail if data validation is not successful. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of information disclosure due to memory buffer overflow attack is minimal.

Denial of Service (Threat #95)

Threat: Attacker attempts to use up all the TPM memory by providing malicious input paramenters.

Mitigation: If the TPM memory does not have enough space to load the object, the command will not execute and an error code will be returned.

Elevation of Privilege (Threat #96)

Threat: Attacker attempts to elevate priviledge vertically by carrying out buffer overflow attacks through this command.

Mitigation: This command validates the input data before executing. If input validation fails, the command will not execute. In addition, priviledge commands will require another authorisation session.

Elevation of Privilege (Threat #159)

Threat: Attacker attempts to elevate priviledge horizontally by attempting to load a child object from a hierarchy that he does not have access to.

Mitigation: Authorisation is required for the use of parent object in this command.

Threats against TPM2_StartAuthSession

Spoofing (Threat #108)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #171)

Threat: Attacker starts an authorisation session and attempt to execute TPM commands.

Mitigation: The session will not continue if the correct authorisation value is not provided to the command that follows.

Spoofing (Threat #180)

Threat: Attacker attempts to change the authorisation value.

Mitigation: The command to change the authorisation value will require the user to provide knowledge of the original authorisation value.

Therefore, the authorisation value has to be stored securely in the TCB. Any user who wishes to access the authorisation value must be authenticated and authorised by the TCB.

Tampering (Threat #109)

Threat: Attacker replays an authorisation session.

Mitigation: Nonces are used in TPM sessions to prevent replay attacks. It is important that the use of the nonce must follow TPM's specifications.

Tampering (Threat #170)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Tampering (Threat #172)

Threat: Attacker edits the session key.

Mitigation: The generation of the session key requires the user application to supply the authorisation value to the object that the user application intends to use in the next TPM command. The salt value used in the generation of the session key is protected cryptographically as well. However, the TCB will have to store the authorisation value securely to prevent misuse.

If the authorisation value is easy to guess, then it is important that an attacker cannot eaves drop on the authorised session as it may be possible to recover the session key.

Repudiation (Threat #110)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #173)

Threat: User application denies starting an authorisation session.

Mitigation: Starting an authorisation session will require the knowledge of the authorisation value. TCB has to securely manage the authorisation value to prevent misuse.

Information Disclosure (Threat #111)

Threat: Attacker steals the session key.

Mitigation: The generation of the session key requires the user application to supply the authorisation value to the object that the user application intends to use in the next TPM command. The salt value used in the generation of the session key is protected cryptographically as well. However, the TCB will have to store the authorisation value securely to prevent misuse.

If the authorisation value is easy to guess, then it is important that an attacker cannot eaves drop on the authorised session as it may be possible to recover the session key.

Information Disclosure (Threat #174)

Threat: Attacker uses physical means to obtain the session key from the TPM.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #175)

Threat: Attacker weakens the cryptographic properties of authorised session by compromisng the salt value used in the generation of the session key.

Mitigation: The method used to generate the salt value has to meet security requirements. Integrity measurement at system startup can prevent unauthorised changes to the salt generating mechanism. However, the TCB will have to put in place security measures to mitigate the risk of run time attacks.

An alternate method is to use TPM's random number generator to provide the salt value. Similarly, TPM's RNG has to meet security requirements.

Denial of Service (Threat #112)

Threat: Attacker attempts to use up all the TPM computing resources by sending this command repeatedly.

Mitigation: TPM will return an error code if there is not enough memory to start this session. TPM does not have feature to control resource consumption.

The integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks.

Elevation of Privilege (Threat #113)

Threat: Attacker attempts to elevate priviledge horizontally by starting an authorisation session to an object that he does not have access to.

Mitigation: When the authorised session is used on another object, the authorisation value to the new object has to be provided. Therefore, authorisation value has to be managed securely to prevent misuse.

Data Flows

Threats against Authorisation data

Tampering (Threat #114)

Threat: Attacker alters data used to start an authorised session.

Mitigation: The salt value is encrypted.

The derivation of the session key requires the authorisation value. This authorisation value is not passed from user application to TPM in the data flow.

Information Disclosure (Threat #115)

Threat: Attacker generates session key based on data sniffed from this data flow.

Mitigation: The salt value is encrypted.

The derivation of the session key requires the authorisation value. This authorisation value is not passed from user application to TPM in the data flow. However, it is important that the authorisation value cannot be guessed easily.

Information Disclosure (Threat #178)

Threat: Attacker tries to read the actual salt value.

Mitigation: The salt value is encrypted with a key loaded in TPM.

Information Disclosure (Threat #179)

Threat: Attacker makes use of physical means to read the key stored in TPM.

Mitigation: The TPM is not designed to withstand physical attacks. However, this risk can be mitigated if the computing system is secured physically.

Denial of Service (Threat #116)

Threat: Attacker hijacks authorisation session.

Mitigation: The establishment of the authorised session will require the knowledge of the authorisation value. Nouces are used to protect against replay attacks.

Threats against Key handle

Tampering (Threat #48)

Threat: Attacker obtains object handle and attempt to carry out a TPM command on that object.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Information Disclosure (Threat #49)

Threat: Attacker obtains object handle and use TPM commands to decrypt the data in the object's private area.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Denial of Service (Threat #50)

Threat: Attacker denies user application from using the object handle.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of object handle.

Threats against Key handle

Tampering (Threat #57)

Threat: Attacker obtains object handle and attempt to carry out a TPM command on that object.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Information Disclosure (Threat #58)

Threat: Attacker obtains object handle and use TPM commands to decrypt the data in the object's private area.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Denial of Service (Threat #59)

Threat: Attacker denies user application from using the object handle.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of object handle.

Threats against Key object ready for migration

Tampering (Threat #60)

Threat: Attacker attempts to edit the duplicated data object.

Mitigation: The data object is encrypted with a key derived from a seed generated by the source TPM. This seed value is encrypted by the public key from the new parent at destination TPM.

Information Disclosure (Threat #61)

Threat: Attacker obtain sensitive information from the duplicated object.

Mitigation: The data object is encrypted with a key derived from a seed generated by the source TPM. This seed value is encrypted by the public key from the new parent at destination TPM.

Denial of Service (Threat #62)

Threat: Attacker denies user access to duplicated object.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of duplicated object.

It will depend on the security of the mechanism used to migrate the duplicated object from the source TPM to the destination TPM.

Threats against Key object

Tampering (Threat #33)

Threat: Attacker attempts to edit the created data object.

Mitigation: If the object is not in a shielded location of a TPM, the integrity and confidentiality of the sensitive area of an object is protected cryptographically.

Information Disclosure (Threat #34)

Threat: The sensitive part of the key object is symmetrically encrypted using a key derived from the parent object. A random value is included in the process as an initialization vector (IV). When an object is created for duplication, the IV is set to zero. The key objects can be susceptible to cryptographic analysis if the parent object is reused multiple times.

Mitigation: The user application has to avoid reusing the parent object multiple times when creating an object for duplication.

Denial of Service (Threat #35)

Threat: Attacker denies user access to created object.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of created object.

Threats against Key object

Tampering (Threat #107)

Threat: Attacker attempts to edit the created data object.

Mitigation: If the object is not in a shielded location of a TPM, the integrity and confidentiality of the sensitive area of an object is protected cryptographically.

Information Disclosure (Threat #120)

Threat: Attacker obtain sensitive information from the created object.

Mitigation: TPM2_Create will encrypt the sensitive area of the object with an encryption key derived from the parent object.

Denial of Service (Threat #47)

Threat: Attacker denies user access to created object.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of created object.

Threats against Session Handle

Tampering (Threat #117)

Threat: Attacker alters the nonce value returned by the TPM.

Mitigation: If the nonce value is altered, the session will not be able to proceed because the nonce value held by the TPM will be different. The nonce value is used in the generation of the HMAC integrity check value and session key.

Tampering (Threat #176)

Threat: Attacker reuse session handle.

Mitigation: Session handle for HMAC or policy authorisation can only be used once. However, session handle for password authorisation can be reused. Hence, the TCB has to securely manage the password to prevent misuse.

Tampering (Threat #181)

Threat: Attacker uses replay attacks to establish a session with TPM.

Mitigation: Nonces are used in TPM sessions to prevent replay attacks. It is important that the use of the nonce must follow TPM's specifications.

Information Disclosure (Threat #118)

Threat: Attacker reads the session handle.

Mitigation: Knowledge of the session handle does not implied that a TPM command does not need to check the authorisation value. If the attacker does not know the authorisation, the TPM command will not execute.

Denial of Service (Threat #119)

Threat: Attacker denies user access to session handle.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of session handle.

Threats against Symmetric key

Tampering (Threat #36)

Threat: An attacker can wire tap the connection between the CPU and TPM and make unauthorised changes to the data exchanges.

Mitigation: TPM sessions can use HMAC to protect the integrity of data exchanges between CPU and TPM. However, the TPM can carry out different type of session (unsalted, unbounded to salted and bounded). Therefore, the user application has to choose an appropriate type of session.

Meanwhile, the attacker will have to gain physical access to the computing system in order to tap the communication bus between the CPU and TPM.

Tampering (Threat #80)

Threat: Attacker insert malicious codes into the data path between the user application and TPM to hijack the session.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

In addition, the session can be encrypted.

Information Disclosure (Threat #37)

Threat: An attacker can wire tap the connection between the CPU and TPM and read the data exchange.

Mitigation: TPM sessions can use symmetric cryptography to protect the confidentiality of data exchanges between CPU and TPM. However, the TPM can carry out different type of session (unsalted, unbounded to salted and bounded). Therefore, the user application has to choose an appropriate type of session.

Meanwhile, the attacker will have to gain physical access to the computing system in order to tap the communication bus between the CPU and TPM.

Information Disclosure (Threat #82)

Threat: Attacker launches physical attacks on TPM to obtain cryptographic key that encrypts the data exchange between the user application and TPM.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #143)

Threat: Attacker insert malicious codes into the data path between the user application and TPM to read the data exchanges.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

Denial of Service (Threat #38)

Threat: Attacker attempts to use up all TPM computing resources by sending large number of commands to TPM.

Mitigation: TPM relies on the TCB to mitigate this type of attack and the integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks. TPM does not control resource consumption but there is dictionary attack protection for authorisation session.

Data Stores

Threats against TPM RAM

Tampering (Threat #63)

Threat: Malicious codes in the computing system attempt to alter the data stored in TPM RAM.

Mitigation: TPM will have to reply on the security of the TCB to stop malicious codes from entering the computing system. The integrity measurement at system start up can protect against this type of threat but there is a risk of run time attacks.

TPM only allow access to data stored in shielded location if the command authorisation is successful. However, the TPM specification allows vendor specific commands to access and modify shielded locations on a TPM under certain circumstances. If such vendor specific commands are considered for implementation, they have to be evaluated to determine whether they meet security requirements.

When an object is loaded into the TPM, the user will be given a handle to reference this object. The TCB has to securely manage the use of this handle and the authorisation value of the parent object to prevent misuse.

The TPM has to rely on the TCB to log down the commands that access the data stored in the TPM.

Repudiation (Threat #64)

Threat: Attacker can access the computing system and edit the log file that records the interactions of the user app with the TPM.

Mitigation: TPM does not store log files internally. It will rely on the TCB to keep logs. Therefore, the TCB has to protect the integrity of the log files and ensure that only authorised changes can be made.

Information Disclosure (Threat #65)

Threat: Malicious codes in computing system attempt to read the data stored in TPM RAM.

Mitigation: TPM will have to reply on the security of the TCB to stop malicious codes from entering the computing system. The integrity measurement at system start up can protect against this type of threat but there is a risk of run time attacks.

Although data can be stored without encryption in TPM RAM, commands that access the data stored in the TPM RAM will require authorisation. The TCB has to ensure that the authorisation value is stored securely.

In addition, TPM RAM will lose the stored data after a restart.

Information Disclosure (Threat #142)

Threat: Attacker makes use of physical means to read data stored in TPM RAM.

Mitigation: The TPM is not designed to withstand physical attacks. However, this risk can be mitigated if the computing system is secured physically.

Denial of Service (Threat #66)

Threat: Attacker attempts to run the TPM RAM out of space.

Mitigation: If there are no more space in TPM RAM, the response code will indicate to the user that there is not enough space to store the object. The TPM should not crash if it runs out of RAM space.

Certifications

No certifications have been made for this threat model.

External Dependencies

ID	Name	URL	Origin	Team Owner	External Owner	Notes
1	TPM device driver		External		TPM manufacturer	
2	Trusted Software Stack		External		TCG	

Implementation Assumptions

ID	Date/Time	Element Impacted	Assumption
1	03/12/2013 4:12:26PM	All	It is assumed that the TPM manufacturer will follow the TPM specification closely.
2	03/14/2013 4:45:56PM	All	It is assumed that the cryptographic standards used by the TPM are robust.
3	03/22/2013 2:48:28PM	All	It is assumed that the internal TPM program codes are secure.

External Security Notes

ID	Notes
1	The Trusted Computing Base has to ensure that authorisation values used by user applications with TPM are managed securely.
2	There must be strong user authentication and authorisation.

Threat Model: User uses TPM 2.0 to decrypt file used for group share

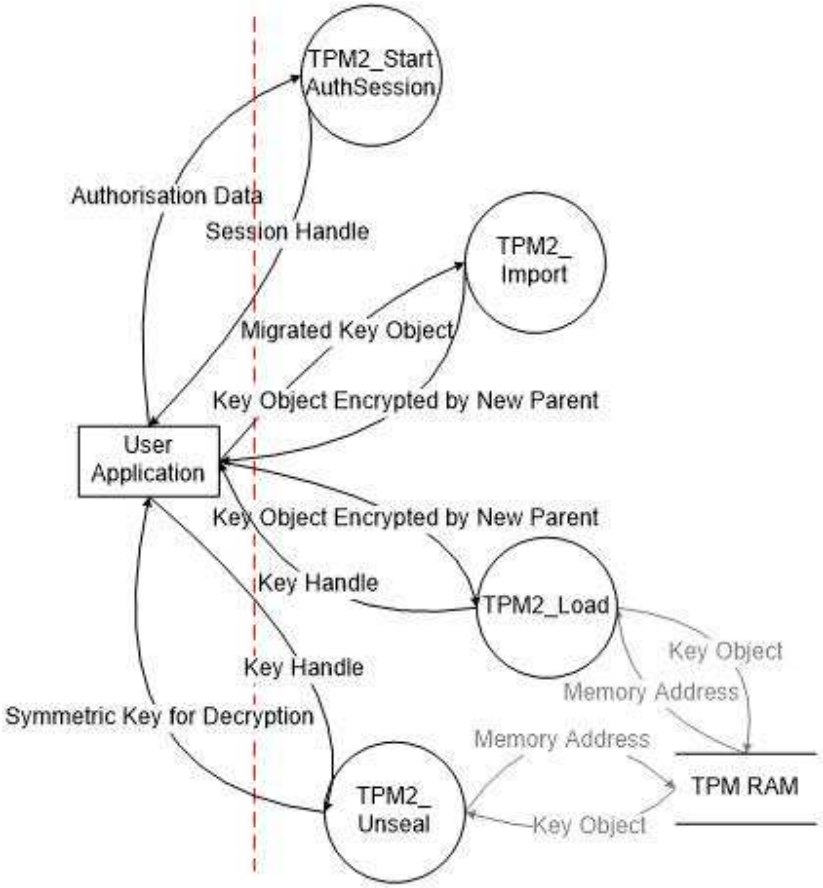
Threat Model Information
Data Flow Diagrams
Threats and Mitigations
Certifications
External Dependencies
Implementation Assumptions
External Security Notes

Threat Model Information

Component	User uses TPM 2.0 to decrypt file used for group share
Product	Trusted Platform Module 2.0
SourceUrl	
Owner	Trusted Computing Group
Participants	Jiun Yi Yap, Allan Tomlinson
Reviewers	
Url	
Summary	A PhD research project at Royal Holloway University of London Information Security Group
History	

Data Flow Diagrams

Context



Threats and Mitigations

Elements

Element Type	Description
Data Flow	Authorisation Data
Data Flow	Key Handle
Data Flow	Key Handle
Data Flow	Key Object Encrypted by New Parent
Data Flow	Key Object Encrypted by New Parent
Data Flow (NotGenerated)	Key Object (This element is not accessible from the entry points.)
Data Flow (NotGenerated)	Key Object (This element is not accessible from the entry points.)

Data Flow (NotGenerated)	Memory Address (This element is not accessible from the entry points.)
Data Flow (NotGenerated)	Memory Address (This element is not accessible from the entry points.)
Data Flow	Migrated Key Object
Data Flow	Session Handle
Data Flow	Symmetric Key for Decryption
Data Store	TPM RAM
Interactor	User Application
Process	TPM2_ Import
Process	TPM2_ Unseal
Process	TPM2_Load
Process	TPM2_StartAuthSession
TrustBoundary	

External Interactors

Threats against User Application

Spoofing (Threat #1)

Threat: Attacker uses social engineering methods to gains access to the computing system and attempt to interact with TPM.

Mitigation: This threat can be mitigated if the use of TPM is combined with more sophisticated authentication methods, eg smart card, 2FA.

Most TPM commands require authorisation. TPM authorised a user based on its knowledge of a secret credential. If the authorisation fails, the user will not be able to carry out the commands. However, the credential has to be stored securely in the TCB.

Spoofing (Threat #71)

Threat: Attacker eavesdrops on authorisation session between user application and TPM.

Mitigation: The salt used to generate the session key is encrypted by a key protected by TPM. The session key in turns encrypt the data exchanges during the session between the user application and TPM.

Spoofing (Threat #72)

Threat: Attacker inserts malicious codes into the computing system and attempt to interact with TPM

Mitigation: TPM will have to reply on the security of the TCB to stop malicious codes from entering the computing system. The integrity measurement at system start up can protect against this type of threat but there is a risk of run time attacks.

Most TPM commands require authorisation. TPM authorised a user based on its knowledge of a secret credential. If the authorisation fails, the user will not be able to carry out the commands. However, the credential has to be stored securely in the TCB.

Spoofing (Threat #73)

Threat: Attacker makes changes to the authorisation value in TPM either by gaining unauthorised access to the computing system or by inserting malicious codes.

Mitigation: Changes to the authorisation value in TPM will require the authorisation of the administrator. However, the administrator credential has to be stored securely in the TCB.

Spoofing (Threat #74)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #111)

Threat: Attacker diverts communication between user application and local TPM to another malicious TPM.

Mitigation: The authorization data is never sent in clear from the user application to TPM. The malicious TPM must know the authorization data before it can trick the user application to believer that it is the local TPM.

Repudiation (Threat #2)

Threat: User denies carrying out certain commands.

Mitigation: TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #75)

Threat: Attacker pretends to be a user application and replays commands.

Mitigation: TPM sessions make use of nouces to mitigate against replay attacks.

Repudiation (Threat #76)

Threat: Attacker accesses the computing system and edits log to misrepresent the record of commands.

Mitigation: Besides strong authenication of user, TCB has to make sure only authorisation changes can be made to the log.

Processes

Threats against TPM2_ Import

Spoofing (Threat #25)

Threat: Attacker attempts to load a migratable key object that is not generated by a TPM.

Mitigation: The source TPM can insert an unique identifying value into the key object when using TPM2_Create. The destination TPM will verify the authencity of the key object by inspecting this identifying value.

Spoofing (Threat #90)

Threat: Attacker pretends to be user application and attempt to execute this command.

Mitigation: The integrity measurement at system start up can detect the presence of unauthorised software. However, there is a risk of executing unauthorised software during run time and the security features of the

TCB will have to mitigate this threat.

The command will check for successful authorisation before executing. However, the credential has to be stored securely in the TCB. In addition, the TCB will have to authenticate the user.

Spoofing (Threat #91)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #92)

Threat: Attacker attempts to change the authorisation value of the new parent.

Mitigation: The command to change the authorisation value will require the user to provide knowledge of the original authorisation value.

Therefore, the authorisation value has to be stored securely in the TCB. Any user who wishes to access the authorisation value must be authenticated and authorised by the TCB.

Tampering (Threat #26)

Threat: Attacker carries out a buffer overflow attack in an attempt to access other data stored in the TPM.

Mitigation: This command validates the input data before executing the process of importing the migrateable key object. If input validation fails, the command will not execute. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of adnormal behaviour due to memory buffer overflow attack is minimal.

Tampering (Threat #94)

Threat: Attacker edits a migrateable key object and attempts to import it.

Mitigation: The migrateable key object has confidentiality and integrity protection provided by cryptographic means. However, there is a risk that the symmetric key for the inner wrapper and seed for the outer wrapper may be compromised. The TCB will have to protect these data by storing them securely and checking the identity and authorisation of the user requesting to access these date.

Tampering (Threat #95)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Repudiation (Threat #27)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent

misuse.

Repudiation (Threat #96)

Threat: User application denies importing this object using the TPM.

Mitigation: Importing the object will require knowledge of the authorisation value of the new parent. TCB will have to check the identity and authorisation of the user if it requests access to the authorisation value.

As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log.

Information Disclosure (Threat #28)

Threat: Attacker attempts to obtain the seed value used to provide the outer layer of encryption by cracking the sharing protocol which uses asymmetric encrpytion.

Mitigation: The public key infrastructure and cryptographic functions used for this purpose have to meet security requirements. The attestation feature of the TPM can also be used to provide some assurance on the authenticity and security of the endpoints.

Information Disclosure (Threat #97)

Threat: Attacker launches physical attacks on TPM to obtain the new parent key in an attempt to break the cryptographic protection for the migrated key object.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #98)

Threat: Attacker carries out crypt analysis on key objects encrypted by the same parent key.

Mitigation: To reduce this risk, the TCB has to ensure that the same parent key must not be reused multiple times. A good method is to continuously expand the object hierarchy so that there are fresh keys to use for encrypting the key object.

Denial of Service (Threat #29)

Threat: Attacker attempts to use up all the TPM computing resources by sending this command repeatedly.

Mitigation: TPM relies on the TCB to mitigate this type of attack. The integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks. TPM does not have any feature to control resource consumption but there is dictionary attack protection for authorisation session.

Denial of Service (Threat #99)

Threat: Attacker launches DOS attack against the migration infrastructure.

Mitigation: TPM has to rely on the infrastructure provider to mitigate this type of attack. If all the computing systems in the network are equipped with TPM, then the integrity measurement and attestation function can provide a higher level of trustworthiness.

Elevation of Privilege (Threat #30)

Threat: Attacker attempts to elevate priviledge vertically by carrying out buffer overflow attacks.

Mitigation: Priviledge commands will require another authorisation session.

Elevation of Privilege (Threat #100)

Threat: Attacker attempts to elevate priviledge horizontally by attempting to duplicate an object that he does not have access to.

Mitigation: The seed to derive the key used to decrypt the outer wrapper is protected by the new parent key. If the attacker tries to import a migrateable object, he must have the authorisation value to this new parent key.

TCB has to securely store this authorisation value and identity and authorisation checking must be carried out if a user request access to this authorisation value.

Threats against TPM2_ Unseal

Spoofing (Threat #37)

Threat: Attacker pretends to be user application and attempt to execute this command.

Mitigation: The integrity measurement at system start up can detect the presence of unauthorised software. However, there is a risk of executing unauthorised software during run time and the security features of the TCB will have to mitigate this threat.

This command is only authorised to be used by the user role and the command will check for successful authorisation before executing. However, the credential has to be stored securely in the TCB. In addition, the TCB will have to authenticate the user.

Spoofing (Threat #102)

Threat: Attacker guesses the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #103)

Threat: Attacker attempts to change the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Tampering (Threat #38)

Threat: Attacker carries out a memory buffer overflow attack in an attempt to access other data stored in the TPM.

Mitigation: This command validates the input data before executing the process of creating the new object. If input validation fails, the command will not execute.

Tampering (Threat #104)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Repudiation (Threat #39)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Information Disclosure (Threat #40)

Threat: Attacker attempts to use this command to recover a cryptographic key which he does not have access to.

Mitigation: The command will not execute if authorisation fails. If the key is generated by TPM, the command will also check if the creator of the key object allows decryption of the cryptographic. In addition, integrity check will detect unauthorised changes to the attributes of the key object.

Denial of Service (Threat #41)

Threat: Attacker attempts to use up all the TPM memory by providing malicious input paramenters.

Mitigation: If the command input refers to an invalid object handle, the command will return an error code.

Elevation of Privilege (Threat #42)

Threat: Attacker attempts to elevate priviledge vertically by carrying out buffer overflow attacks through this command.

Mitigation: This command validates the input data before executing. If input validation fails, the command will not execute. In addition, priviledge commands will require another authorisation session.

Elevation of Privilege (Threat #105)

Threat: Attacker attempts to elevate priviledge horizontally by attempting to unseal a data object that he does not have access to.

Mitigation: The provided authorisation value has to match that of the data object. If there is no match, the command will not execute.

Threats against TPM2_Load

Spoofing (Threat #31)

Threat: Attacker pretends to be user application and attempt to execute this command.

Mitigation: The integrity measurement at system start up can detect the presence of unauthorised software. However, there is a risk of executing unauthorised software during run time and the security features of the TCB will have to mitigate this threat.

This command is only authorised to be used by the user role and the command will check for successful authorisation before executing. However, the credential has to be stored securely in the TCB. In addition, the TCB will have to authenticate the user.

Spoofing (Threat #77)

Threat: Attacker guesses the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #78)

Threat: Attacker creates a TPM object externally and attempts to load this object into TPM.

Mitigation: This command will check the integrity, attributes and cryptographic binding of the object before loading it into the TPM. The command will not execute and it return an error code if the check fails.

Tampering (Threat #32)

Threat: Attacker carries out a memory buffer overflow attack in an attempt to access other data stored in the TPM.

Mitigation: This command validates the input data before executing the process of loading the new object. If input validation fails, the command will not execute. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of adnormal behaviour due to memory buffer overflow attack is minimal.

Tampering (Threat #79)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Tampering (Threat #80)

Threat: Attacker edits an object and attempt to load it into TPM.

Mitigation: This command will check the integrity, attributes and cryptographic binding of the object before loading it into the TPM. The command will not execute and it return an error code if the check fails.

Repudiation (Threat #33)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #81)

Threat: User application denies loading this object to the TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Information Disclosure (Threat #34)

Threat: Attacker carries out memory buffer overflow attacks in an attempt to access data stored in TPM.

Mitigation: This command carries out validation of input data. The command will fail if data validation is not successful. In addition, TPM only allows a protected capability to access data in a shield location. Data outside a shielded location has its integrity and confidentiality protected cryptographically. Hence, the risk of information disclosure due to memory buffer overflow attack is minimal.

Denial of Service (Threat #35)

Threat: Attacker attempts to use up all the TPM memory by providing malicious input paramenters.

Mitigation: If the TPM memory does not have enough space to load the object, the command will not execute and an error code will be returned.

Elevation of Privilege (Threat #36)

Threat: Attacker attempts to elevate priviledge vertically by carrying out buffer overflow attacks through this command.

Mitigation: This command validates the input data before executing. If input validation fails, the command will not execute. In addition, priviledge commands will require another authorisation session.

Elevation of Privilege (Threat #82)

Threat: Attacker attempts to elevate priviledge horizontally by attempting to load a child object from a hierarchy that he does not have access to.

Mitigation: Authorisation is required for the use of parent object in this command.

Threats against TPM2_StartAuthSession

Spoofing (Threat #15)

Threat: Attacker guess the authorisation value.

Mitigation: TPM has a dictionary attack protection feature and will enter a lock out mode when there is successive authorization failure. However, the authorisation value must not be of a value that can be guessed easily.

Spoofing (Threat #83)

Threat: Attacker starts an authorisation session and attempt to execute TPM commands.

Mitigation: The session will not continue if the correct authorisation value is not provided to the command that follows.

Spoofing (Threat #93)

Threat: Attacker attempts to change the authorisation value.

Mitigation: The command to change the authorisation value will require the user to provide knowledge of the original authorisation value.

Therefore, the authorisation value has to be stored securely in the TCB. Any user who wishes to access the authorisation value must be authenticated and authorised by the TCB.

Tampering (Threat #16)

Threat: Attacker replays an authorisation session.

Mitigation: Nonces are used in TPM sessions to prevent replay attacks. It is important that the use of the nonce must follow TPM's specifications.

Tampering (Threat #84)

Threat: Attacker attempts to change the behaviour of this command by changing TPM firmware to an unauthorised version.

Mitigation: During field upgrade process, the TCB has to carry out integrity check on the new firmware and ensure that the firmware is endorsed by the manufacturer.

Tampering (Threat #85)

Threat: Attacker edits the session key.

Mitigation: The generation of the session key requires the user application to supply the authorisation value to the object that the user application intends to use in the next TPM command. The salt value used in the generation of the session key is protected cryptographically as well. However, the TCB will have to store the authorisation value securely to prevent misuse.

If the authorisation value is easy to guess, then it is important that an attacker cannot eaves drop on the authorised session as it may be possible to recover the session key.

Repudiation (Threat #17)

Threat: User application denies executing this command on TPM.

Mitigation: As the TPM does not keep an internal log, the TCB will have to keep a log of the commands performed on TPM. In addition, TCB will have to protect the integrity of this log and only allows authorised changes to be made to the log. Meanwhile, TCB has to store the authorisation values securely to prevent misuse.

Repudiation (Threat #86)

Threat: User application denies starting an authorisation session.

Mitigation: Starting an authorisation session will require the knowledge of the authorisation value. TCB has to securely manage the authorisation value to prevent misuse.

Information Disclosure (Threat #18)

Threat: Attacker steals the session key.

Mitigation: The generation of the session key requires the user application to supply the authorisation value to the object that the user application intends to use in the next TPM command. The salt value used in the generation of the session key is protected cryptographically as well. However, the TCB will have to store the authorisation value securely to prevent misuse.

If the authorisation value is easy to guess, then it is important that an attacker cannot eaves drop on the authorised session as it may be possible to recover the session key.

Information Disclosure (Threat #87)

Threat: Attacker uses physical means to obtain the session key from the TPM.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #88)

Threat: The cryptographic properties of authorised session can be weaken if the nonce and salt value used in the generation of the session key have low entropy.

Mitigation: The method used to generate the nonce and salt value has to meet security requirements, for example NIST SP 800-90A. An alternate method is to use TPM's random number generator to provide the salt value. Similarly, TPM's RNG has to meet security requirements.

Denial of Service (Threat #19)

Threat: Attacker attempts to use up all the TPM computing resources by sending this command repeatedly.

Mitigation: TPM will return an error code if there is not enough memory to start this session. TPM does not have feature to control resource consumption.

The integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks.

Elevation of Privilege (Threat #20)

Threat: Attacker attempts to elevate priviledge horizontally by starting an authorisation session to an object that he does not have access to.

Mitigation: When the authorised session is used on another object, the authorisation value to the new object has to be provided. Therefore, authorisation value has to be managed securely to prevent misuse.

Data Flows

Threats against Authorisation Data

Tampering (Threat #3)

Threat: Attacker alters data used to start an authorised session.

Mitigation: The salt value is encrypted.

The derivation of the session key requires the authorisation value. This authorisation value is not passed from user application to TPM in the data flow.

Information Disclosure (Threat #4)

Threat: Attacker generates session key based on data sniffed from this data flow.

Mitigation: The salt value is encrypted.

The derivation of the session key requires the authorisation value. This authorisation value is not passed from user application to TPM in the data flow. However, it is important that the authorisation value cannot be guessed easily.

Information Disclosure (Threat #68)

Threat: Attacker tries to read the actual salt value.

Mitigation: The salt value is encrypted with a key loaded in TPM.

Information Disclosure (Threat #69)

Threat: Attacker makes use of physical means to read the key stored in TPM.

Mitigation: The TPM is not designed to withstand physical attacks. However, this risk can be mitigated if the computing system is secured physically.

Denial of Service (Threat #5)

Threat: Attacker hijacks authorisation session.

Mitigation: The establishment of the authorised session will require the knowledge of the authorisation value. Nouces are used to protect against replay attacks.

Threats against Key Handle

Tampering (Threat #52)

Threat: Attacker obtains object handle and attempt to carry out a TPM command on that object.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Information Disclosure (Threat #53)

Threat: Attacker obtains object handle and use TPM commands to decrypt the data in the object's private area.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Denial of Service (Threat #54)

Threat: Attacker denies user application from using the object handle.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of object handle.

Threats against Key Handle

Tampering (Threat #55)

Threat: Attacker obtains object handle and attempt to carry out a TPM command on that object.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Information Disclosure (Threat #56)

Threat: Attacker obtains object handle and use TPM commands to decrypt the data in the object's private area.

Mitigation: TPM command will check that the authorisation value provided by the user for the object matches the authorisation value stored in the TPM. The command will only operate on the object if the authorisation is

successful. Hence, the TCB has to securely manage the authorisation value to prevent misuse.

Denial of Service (Threat #57)

Threat: Attacker denies user application from using the object handle.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of object handle.

Threats against Key Object Encrypted by New Parent

Tampering (Threat #46)

Threat: Attacker attempts to edit the key object.

Mitigation: The key object is cryptographically protected by the new parent.

Information Disclosure (Threat #47)

Threat: Attacker obtain sensitive information from the key object.

Mitigation: The key object is cryptographically protected by the new parent.

Denial of Service (Threat #48)

Threat: Attacker denies user access to key object.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of created object.

Threats against Key Object Encrypted by New Parent

Tampering (Threat #49)

Threat: Attacker attempts to edit the key object.

Mitigation: The key object is cryptographically protected by the new parent.

Information Disclosure (Threat #50)

Threat: Attacker obtain sensitive information from the key object.

Mitigation: The key object is cryptographically protected by the new parent.

Denial of Service (Threat #51)

Threat: Attacker denies user access to key object.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of created object.

Threats against Migrated Key Object

Tampering (Threat #43)

Threat: Attacker attempts to edit the migrated data object.

Mitigation: The data object is encrypted with a key derived from a seed generated by the source TPM. This seed value is encrypted by the public key from the new parent at destination TPM.

Information Disclosure (Threat #44)

Threat: Attacker obtain sensitive information from the migrated object.

Mitigation: The data object is encrypted with a key derived from a seed generated by the source TPM. This seed value is encrypted by the public key from the new parent at destination TPM.

Denial of Service (Threat #45)

Threat: Attacker denies user access to migrated object.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of duplicated object.

It will depend on the security of the mechanism used to migrate the duplicated object from the source TPM to the destination TPM.

Threats against Session Handle

Tampering (Threat #6)

Threat: Attacker alters the nonce value returned by the TPM.

Mitigation: If the nonce value is altered, the session will not be able to proceed because the nonce value held by the TPM will be different. The nonce value is used in the generation of the HMAC integrity check value and session key.

Tampering (Threat #89)

Threat: Attacker reuse session handle.

Mitigation: Session handle for HMAC or policy authorisation can only be used once. However, session handle for password authorisation can be reused. Hence, the TCB has to securely manage the password to prevent misuse.

Information Disclosure (Threat #7)

Threat: Attacker reads the session handle.

Mitigation: Knowledge of the session handle does not implied that a TPM command does not need to check the authorisation value. If the attacker does not know the authorisation, the TPM command will not execute.

Denial of Service (Threat #8)

Threat: Attacker denies user access to session handle.

Mitigation: Integrity measurement at system start up can check for the presence of malicious software.

Strong authentication and authorisation must be used to prevent misuse of session handle.

Threats against Symmetric Key for Decryption

Tampering (Threat #58)

Threat: An attacker can wire tap the connection between the CPU and TPM and make unauthorised changes to the data exchanges.

Mitigation: TPM sessions can use HMAC to protect the integrity of data exchanges between CPU and TPM. However, the TPM can carry out different type of session (unsalted, unbounded to salted and bounded). Therefore, the user application has to choose an appropriate type of session.

Meanwhile, the attacker will have to gain physical access to the computing system in order to tap the communication bus between the CPU and TPM.

Tampering (Threat #106)

Threat: Attacker insert malicious codes into the data path between the user application and TPM to hijack the session.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

Information Disclosure (Threat #59)

Threat: Attacker tries to obtain the session key that is used to encrypt the return value.

Mitigation: The TCB has to put in place security mechanism to protect the session key it uses with the TPM. When the return value leaves the TPM, the TCB has to store the return value securely. User access to the return value should be authenticated and check for authorisation.

Information Disclosure (Threat #107)

Threat: An attacker can wire tap the connection between the CPU and TPM and read the data exchange.

Mitigation: TPM sessions can use symmetric cryptography to protect the confidentiality of data exchanges between CPU and TPM. However, the TPM can carry out different type of session (unsalted, unbounded to salted and bounded). Therefore, the user application has to choose an appropriate type of session.

Meanwhile, the attacker will have to gain physical access to the computing system in order to tap the communication bus between the CPU and TPM.

Information Disclosure (Threat #108)

Threat: Attacker launches physical attacks on TPM to obtain cryptographic key that encrypts the data exchange between the user application and TPM.

Mitigation: TPM is not designed to withstand physical attacks. User can consider using physical measures (eg. a lock) to prevent unauthorised physical access to computing system.

Information Disclosure (Threat #109)

Threat: Attacker insert malicious codes into the data path between the user application and TPM to read the data exchanges.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

Denial of Service (Threat #60)

Threat: Attacker attempts to use up all TPM computing resources by sending large number of commands to TPM.

Mitigation: TPM relies on the TCB to mitigate this type of attack and the integrity measurement at system start up ensure that the TCB is not compromised by malicious software. However, the TCB will need to be protected against run time attacks. TPM does not control resource consumption but there is dictionary attack protection for authorisation session.

Data Stores

Threats against TPM RAM

Tampering (Threat #21)

Threat: Malicious codes in the computing system attempt to alter the data stored in TPM RAM.

Mitigation: TPM will have to reply on the security of the TCB to stop malicious codes from entering the computing system. The integrity measurement at system start up can protect against this type of threat but there is a risk of run time attacks.

TPM only allow access to data stored in shielded location if the command authorisation is successful. However, the TPM specification allows vendor specific commands to access and modify shielded locations on a TPM under certain circumstances. If such vendor specific commands are considered for implementation, they have to be evaluated to determine whether they meet security requirements.

When an object is loaded into the TPM, the user will be given a handle to reference this object. The TCB has to securely manage the use of this handle and the authorisation value of the parent object to prevent misuse.

The TPM has to rely on the TCB to log down the commands that access the data stored in the TPM.

Repudiation (Threat #22)

Threat: Attacker can access the computing system and edit the log file that records the interactions of the user app with the TPM.

Mitigation: TPM does not store log files internally. It will rely on the TCB to keep logs. Therefore, the TCB has to protect the integrity of the log files and ensure that only authorised changes can be made.

Information Disclosure (Threat #23)

Threat: Malicious codes in computing system attempt to read the data stored in TPM RAM.

Mitigation: TPM will have to reply on the security of the TCB to stop malicious codes from entering the computing system. The integrity measurement at system start up can protect against this type of threat but there is a risk of run time attacks.

Although data can be stored without encryption in TPM RAM, commands that access the data stored in the TPM RAM will require authorisation. The TCB has to ensure that the authorisation value is stored securely.

In addition, TPM RAM will lose the stored data after a restart.

Information Disclosure (Threat #70)

Threat: Attacker makes use of physical means to read data stored in TPM RAM.

Mitigation: The TPM is not designed to withstand physical attacks. However, this risk can be mitigated if the

computing system is secured physically.

Denial of Service (Threat #24)

Threat: Attacker attempts to run the TPM RAM out of space.

Mitigation: If there are no more space in TPM RAM, the response code will indicate to the user that there is not enough space to store the object. The TPM should not crash if it runs out of RAM space.

Certifications

No certifications have been made for this threat model.

External Dependencies

ID	Name	URL	Origin	Team Owner	External Owner	Notes
1	TPM device driver		External		TPM manufacturer	
2	Trusted Software Stack		External		TCG	

Implementation Assumptions

ID	Date/Time	Element Impacted	Assumption
1	04/08/2013 2:59:12PM	All	It is assumed that the TPM manufacturer will follow the TPM specification closely.
2	04/08/2013 2:59:26PM	All	It is assumed that the cryptographic standards used by the TPM are robust.
3	04/08/2013 2:59:45PM	All	It is assumed that the internal TPM program codes are secure.

External Security Notes

ID	Notes
1	The Trusted Computing Base has to ensure that authorisation values used by user applications with TPM are managed securely.
2	There must be strong user authentication and authorisation.

Appendix C

An Ontology of a Computing Device Secured with Trusted Platform Module 2.0 in OWL Format

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY tpm2.0 "http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#" >
]>


<rdf:RDF xmlns="http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#"
     xml:base="http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:tpm2.0="http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#">
    <owl:Ontology rdf:about="http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0"/>



    <!--

///////////////////////////////////////////////////////////////
//////////////////////
    //
    // Object Properties
    //

///////////////////////////////////////////////////////////////
//////////////////////
     -->




    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#backsSecurityAndTrustNotion -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;backsSecurityAndTrustNotion">
        <rdfs:domain rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:range
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
    </owl:ObjectProperty>
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#enablesDeviceCapability -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;enablesDeviceCapability">
        <rdfs:range rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:domain rdf:resource="&tpm2.0;TPM2.0Capability"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#isBackedByDeviceCapability -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;isBackedByDeviceCapability">
        <rdfs:range rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:domain
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <owl:inverseOf
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#isCounterMeasureTo -->

    <owl:ObjectProperty rdf:about="&tpm2.0;isCounterMeasureTo">
        <rdfs:domain rdf:resource="&tpm2.0;TPM2.0Capability"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#isEnabledByTPM2.0Capability -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;isEnabledByTPM2.0Capability">
        <rdfs:domain rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:range rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <owl:inverseOf
rdf:resource="&tpm2.0;enablesDeviceCapability"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#isSupportedByTPM2.0Capability -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;isSupportedByTPM2.0Capability">
        <rdfs:range rdf:resource="&tpm2.0;TPM2.0Capability"/>
```

```
        <rdfs:domain
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <owl:inverseOf
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#supportsTPM2.0Capability -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;supportsTPM2.0Capability">
        <rdfs:range rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:domain rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <owl:inverseOf
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#supportsSecurityAndTrustNotion -->

    <owl:ObjectProperty
rdf:about="&tpm2.0;supportsSecurityAndTrustNotion">
        <rdfs:domain rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:range
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#usesTPM2.0SubSystem -->

    <owl:ObjectProperty rdf:about="&tpm2.0;usesTPM2.0SubSystem">
        <rdfs:domain rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:range rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
    </owl:ObjectProperty>



    <!--

///////////////////////////////////////////////////////////////
//////////////////////
    //
    // Classes
    //

///////////////////////////////////////////////////////////////
//////////////////////
     -->
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#AES -->

    <owl:Class rdf:about="&tpm2.0;AES">
        <rdfs:subClassOf rdf:resource="&tpm2.0;SymmetricEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#AccessControl -->

    <owl:Class rdf:about="&tpm2.0;AccessControl">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Integrity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Accountability"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Availability"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Confidentiality"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
```

```
rdf:resource="&tpm2.0;isEnabledByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Accountability -->

    <owl:Class rdf:about="&tpm2.0;Accountability">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IdentityManagement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;AccessControl"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Assurance -->

    <owl:Class rdf:about="&tpm2.0;Assurance">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
```

```
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#AsymmetricEngine -->

    <owl:Class rdf:about="&tpm2.0;AsymmetricEngine">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Certification"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Attestation -->

    <owl:Class rdf:about="&tpm2.0;Attestation">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;AsymmetricEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ExecutionEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
```

```
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Identity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;NVMemory"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ExpectedBehaviour"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authorization"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authenticity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;VolatileMemory"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;KeyGeneration"/>
```

```
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;enablesDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IdentityManagement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Authenticity -->

    <owl:Class rdf:about="&tpm2.0;Authenticity">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Certification"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IdentityManagement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Authorization -->

    <owl:Class rdf:about="&tpm2.0;Authorization">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
```

```
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Certification"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Availability -->

    <owl:Class rdf:about="&tpm2.0;Availability">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;SecureDataBackup"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
```

```
rdf:resource="&tpm2.0;AccessControl"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Certification -->

    <owl:Class rdf:about="&tpm2.0;Certification">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;NVMemory"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;AsymmetricEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Identity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ExecutionEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authenticity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
```

```
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authorization"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Confidentiality -->

    <owl:Class rdf:about="&tpm2.0;Confidentiality">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;SecureDataBackup"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;AccessControl"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#DeviceCapability -->

    <owl:Class rdf:about="&tpm2.0;DeviceCapability"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#ECC -->
```

```
    <owl:Class rdf:about="&tpm2.0;ECC">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;AsymmetricEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#ExecutionEngine -->

    <owl:Class rdf:about="&tpm2.0;ExecutionEngine">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Certification"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#ExpectedBehaviour -->

    <owl:Class rdf:about="&tpm2.0;ExpectedBehaviour">
        <rdfs:subClassOf
```

```
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;SecureDataBackup"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#HashEngine -->

    <owl:Class rdf:about="&tpm2.0;HashEngine">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Identity -->

    <owl:Class rdf:about="&tpm2.0;Identity">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
```

```
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IdentityManagement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Certification"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#IdentityManagement -->

    <owl:Class rdf:about="&tpm2.0;IdentityManagement">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isEnabledByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isEnabledByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isEnabledByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
```

```
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Identity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Accountability"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authenticity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Integrity -->

    <owl:Class rdf:about="&tpm2.0;Integrity">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isSupportedByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isBackedByDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;AccessControl"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#IntegrityMeasurement -->

    <owl:Class rdf:about="&tpm2.0;IntegrityMeasurement">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authorization"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Accountability"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;VolatileMemory"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;HashEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Integrity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;enablesDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IdentityManagement"/>
```

```
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ExecutionEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ExpectedBehaviour"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#KeyGeneration -->

    <owl:Class rdf:about="&tpm2.0;KeyGeneration">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Management -->

    <owl:Class rdf:about="&tpm2.0;Management">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
    </owl:Class>
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#NVMemory -->

    <owl:Class rdf:about="&tpm2.0;NVMemory">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Certification"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#OneTimePadXOR -->

    <owl:Class rdf:about="&tpm2.0;OneTimePadXOR">
        <rdfs:subClassOf rdf:resource="&tpm2.0;SymmetricEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#PowerDetection -->

    <owl:Class rdf:about="&tpm2.0;PowerDetection">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
```

```
000/ontologies/2015/3/TPM2.0#ProtectedLocation -->

    <owl:Class rdf:about="&tpm2.0;ProtectedLocation">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;TPM2.0Capability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;enablesDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;AccessControl"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Authorization"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Confidentiality"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ExecutionEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom rdf:resource="&tpm2.0;RNG"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;enablesDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;SecureDataBackup"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
```

```
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;KeyGeneration"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;enablesDeviceCapability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IdentityManagement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;SymmetricEngine"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;usesTPM2.0SubSystem"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;NVMemory"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#RNG -->

    <owl:Class rdf:about="&tpm2.0;RNG">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#RSA -->
```

```
    <owl:Class rdf:about="&tpm2.0;RSA">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;AsymmetricEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SHA-1 -->

    <owl:Class rdf:about="&tpm2.0;SHA-1">
        <rdfs:subClassOf rdf:resource="&tpm2.0;HashEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SHA-2 -->

    <owl:Class rdf:about="&tpm2.0;SHA-2">
        <rdfs:subClassOf rdf:resource="&tpm2.0;HashEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SM2 -->

    <owl:Class rdf:about="&tpm2.0;SM2">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;AsymmetricEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SM3 -->

    <owl:Class rdf:about="&tpm2.0;SM3">
        <rdfs:subClassOf rdf:resource="&tpm2.0;HashEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SM4 -->

    <owl:Class rdf:about="&tpm2.0;SM4">
        <rdfs:subClassOf rdf:resource="&tpm2.0;SymmetricEngine"/>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
```

```
000/ontologies/2015/3/TPM2.0#SecureDataBackup -->

    <owl:Class rdf:about="&tpm2.0;SecureDataBackup">
        <rdfs:subClassOf
rdf:resource="&tpm2.0;DeviceCapability"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;isEnabledByTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Availability"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;backsSecurityAndTrustNotion"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Confidentiality"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SymmetricEngine -->

    <owl:Class rdf:about="&tpm2.0;SymmetricEngine">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;ProtectedLocation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#TPM2.0Capability -->

    <owl:Class rdf:about="&tpm2.0;TPM2.0Capability"/>
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#TPM2.0SubSystem -->

    <owl:Class rdf:about="&tpm2.0;TPM2.0SubSystem"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SecurityAndTrustNotion -->

    <owl:Class rdf:about="&tpm2.0;SecurityAndTrustNotion"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#VolatileMemory -->

    <owl:Class rdf:about="&tpm2.0;VolatileMemory">
        <rdfs:subClassOf rdf:resource="&tpm2.0;TPM2.0SubSystem"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;Attestation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="&tpm2.0;supportsTPM2.0Capability"/>
                <owl:someValuesFrom
rdf:resource="&tpm2.0;IntegrityMeasurement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>



    <!--

////////////////////////////////////////////////////////////////
//////////////////////
    //
    // Individuals
    //

////////////////////////////////////////////////////////////////
//////////////////////
     -->
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#AES -->

    <owl:NamedIndividual rdf:about="&tpm2.0;AES">
        <rdf:type rdf:resource="&tpm2.0;SymmetricEngine"/>
    </owl:NamedIndividual>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Accountability -->

    <owl:NamedIndividual rdf:about="&tpm2.0;Accountability"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Authenticity -->

    <owl:NamedIndividual rdf:about="&tpm2.0;Authenticity"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Availability -->

    <owl:NamedIndividual rdf:about="&tpm2.0;Availability"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Confidentiality -->

    <owl:NamedIndividual rdf:about="&tpm2.0;Confidentiality"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#ECC -->

    <owl:NamedIndividual rdf:about="&tpm2.0;ECC"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#EndorsementSeed -->

    <owl:NamedIndividual rdf:about="&tpm2.0;EndorsementSeed">
        <rdf:type rdf:resource="&tpm2.0;NVMemory"/>
    </owl:NamedIndividual>
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#ExpectedBehaviour -->

    <owl:NamedIndividual rdf:about="&tpm2.0;ExpectedBehaviour"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Identity -->

    <owl:NamedIndividual rdf:about="&tpm2.0;Identity"/>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#Integrity -->

    <owl:NamedIndividual rdf:about="&tpm2.0;Integrity">
        <rdf:type rdf:resource="&tpm2.0;SecurityAndTrustNotion"/>
    </owl:NamedIndividual>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#OneTimePadXOR -->

    <owl:NamedIndividual rdf:about="&tpm2.0;OneTimePadXOR">
        <rdf:type rdf:resource="&tpm2.0;SymmetricEngine"/>
    </owl:NamedIndividual>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#PCR -->

    <owl:NamedIndividual rdf:about="&tpm2.0;PCR">
        <rdf:type rdf:resource="&tpm2.0;VolatileMemory"/>
    </owl:NamedIndividual>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#PlatformSeed -->

    <owl:NamedIndividual rdf:about="&tpm2.0;PlatformSeed">
        <rdf:type rdf:resource="&tpm2.0;NVMemory"/>
    </owl:NamedIndividual>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#RSA -->
```

```
    <owl:NamedIndividual rdf:about="&tpm2.0;RSA"/>


    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SHA-1 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;SHA-1">
        <rdf:type rdf:resource="&tpm2.0;HashEngine"/>
    </owl:NamedIndividual>


    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SHA-256 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;SHA-256">
        <rdf:type rdf:resource="&tpm2.0;HashEngine"/>
    </owl:NamedIndividual>


    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SM2 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;SM2"/>


    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SM3 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;SM3">
        <rdf:type rdf:resource="&tpm2.0;HashEngine"/>
    </owl:NamedIndividual>


    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#SM4 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;SM4">
        <rdf:type rdf:resource="&tpm2.0;SymmetricEngine"/>
    </owl:NamedIndividual>


    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#StorageSeed -->

    <owl:NamedIndividual rdf:about="&tpm2.0;StorageSeed">
        <rdf:type rdf:resource="&tpm2.0;NVMemory"/>
    </owl:NamedIndividual>
```

```
    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#key_size_1024 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;key_size_1024">
        <rdf:type rdf:resource="&tpm2.0;RSA"/>
    </owl:NamedIndividual>



    <!-- http://www.semanticweb.org/pxai0_
000/ontologies/2015/3/TPM2.0#key_size_2048 -->

    <owl:NamedIndividual rdf:about="&tpm2.0;key_size_2048">
        <rdf:type rdf:resource="&tpm2.0;RSA"/>
    </owl:NamedIndividual>
</rdf:RDF>



<!-- Generated by the OWL API (version 3.5.1)
http://owlapi.sourceforge.net -->
```

Appendix D

Threat Modelling Report of the Use Scenario in Chapter 6

Threat Model: provenance aware system, provenance record store, trust evaluator and rule store

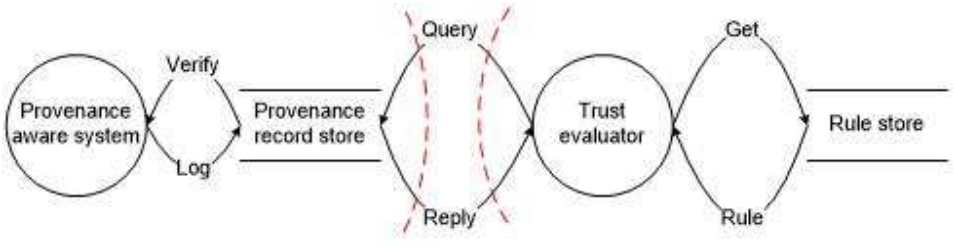
Threat Model Information
Certifications
External Security Notes
Data Flow Diagrams
External Dependencies
Threats and Mitigations
Implementation Assumptions

Threat Model Information

Component	provenance aware system, provenance record store, trust evaluator and rule store
Product	Provenance-based Attestation
SourceUrl	
Owner	Jiun Yi Yap
Participants	Jiun Yi Yap, Allan Tomlinson
Reviewers	
Url	
Summary	A PhD research project at Royal Holloway University of London Information Security Group
History	

Data Flow Diagrams

Context



Threats and Mitigations

Elements

Element Type	Description
Data Flow	Get
Data Flow	Log
Data Flow	Query
Data Flow	Reply
Data Flow	Rule
Data Flow	Verify
Data Store	Provenance record store
Data Store	Rule store
Process	Provenance aware system
Process	Trust evaluator
TrustBoundary	
TrustBoundary	

Processes

Threats against Provenance aware system

Spoofing (Threat #82)

Threat: Attacker pretends to be a legitimate provenance aware system

Mitigation: The trust evaluator does not limit the check to provenance record. It can check on other evidence to ascertain the identity of the provenance aware system.

Tampering (Threat #83)

Threat: Attacker tamper with a legitimate provenance aware system.

Mitigation: A trusted computing base secured with anti-virus software and intrusion detection system can help to mitigate this threat.

Repudiation (Threat #84)

Threat: Provenance aware system denies it produces a particular provenance record.

Mitigation: A provenance aware system can have an unique ID and this ID is tagged to the provenance record it produces.

Information Disclosure (Threat #85)

Threat: Attacker reads provenance records produced by provenance aware system.

Mitigation: Most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Denial of Service (Threat #86)

Threat: Attacker overloads a provenance aware system.

Mitigation: A firewall can block out access from unauthorised IP addresses.

Elevation of Privilege (Threat #87)

Threat: Attacker gain access to high privilege level of provenance aware system.

Mitigation: Most modern operating system requires authorisation before giving a user a higher privilege level. In addition, most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Threats against Trust evaluator

Spoofing (Threat #31)

Threat: Malicious application pretends to be evaluator and gives the result of a positive trust assessment of a provenance record.

Mitigation: The result of the evaluation contains an unique key and it can be used with TPM 2.0 enhanced authorisation.

Tampering (Threat #32)

Threat: The evaluator is modified so that it gives an unreliable assessment of provenance record.

Mitigation: The evaluator should be part of the Trusted Computing Base and its integrity is checked during system boot up.

Repudiation (Threat #33)

Threat: The evaluator denies it's ownership of the assessment of a provenance record.

Mitigation: The evaluator is configured with an unique ID and this ID is stated in the assessment result it produces.

Information Disclosure (Threat #34)

Threat: Attacker reads the input and output of the trust evaluator.

Mitigation: Most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Denial of Service (Threat #35)

Threat: Attacker overloads the computing resource of the trust evaluator.

Mitigation: A firewall can block access from unauthorised IP addresses.

Elevation of Privilege (Threat #36)

Threat: Attacker gains access to higher privilege level of trust evaluator.

Mitigation: Most modern operating system requires authorisation before giving a user a higher privilege level. In addition, most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Data Flows

Threats against Get

Tampering (Threat #47)

Threat: Attacker alters the get rule request from the evaluator to the rule base

Mitigation: Most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Information Disclosure (Threat #48)

Threat: Attacker reads the get rule request from the evaluator to the rule base

Mitigation: Most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Denial of Service (Threat #49)

Threat: Attackers attempt to use up all the computing resource of the rule store by sending larger number of get rule request to the rule base.

Mitigation: If the rule store is accessed over a network, a firewall can be used to block network access from unauthorised ip address.

Threats against Log

Tampering (Threat #9)

Threat: Attacker insert malicious codes into the data path between the provenance aware application and provenance record store to hijack the session.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

In addition, most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Information Disclosure (Threat #10)

Threat: Attacker reads the data path between the provenance aware application and provenance record store.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

In addition, most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Denial of Service (Threat #11)

Threat: Attacker attempts to use up all computing resources by sending large number of logs to the provenance record store.

Mitigation: A provenance aware application has to log in to the provenance record store before it can deposit its provenance record. The log in access control can prevent a provenance aware application from logging in after a number of unsuccessful attempts.

Threats against Query

Tampering (Threat #37)

Threat: Attacker alters the query from the evaluator to the provenance record store.

Mitigation: Network protocol such as the TLS can be used to protect this information exchange.

Information Disclosure (Threat #38)

Threat: Attacker reads the information exchange between the provenance record store and evaluator.

Mitigation: The provenance record is encrypted and the cryptographic key is stored in the source TPM. During the migration of the cryptographic key to the destination TPM, the cryptographic key is encrypted by the public key of the destination TPM.

Meanwhile, network protocol such as the TLS can be used to further protect this information exchange.

Information Disclosure (Threat #75)

Threat: Attacker pretends to be an evaluator and sends a query to the computing device.

Mitigation: The evaluator provides its credentials to the computing device for verification. The credentials can be signed by its TPM.

Denial of Service (Threat #39)

Threat: Attacker attempts to use up the computing resource of the provenance record store by sending large number of queries.

Mitigation: If the provenance record store is accessed over a network, a firewall can be used to block network access from unauthorised ip address.

Threats against Reply

Tampering (Threat #40)

Threat: Attacker alters the reply from the provenance record store to the evaluator.

Mitigation: The provenance record is encrypted and the cryptographic key used is encrypted with the public key of the evaluator's TPM.

Information Disclosure (Threat #41)

Threat: Attacker reads the reply from the provenance record store to the evaluator.

Mitigation: The provenance record is encrypted and the cryptographic key used is encrypted with the public key of the evaluator's TPM.

Denial of Service (Threat #42)

Threat: Attacker attempts to use up the computing resource of the evaluator by sending large number of replies.

Mitigation: A firewall can be used to block network access from unauthorised ip address.

Threats against Rule

Tampering (Threat #50)

Threat: Attacker alters the rule provided by the rule base to the evaluator.

Mitigation: Most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Information Disclosure (Threat #51)

Threat: Attacker reads the rule provided by the rule base to the evaluator.

Mitigation: Most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Denial of Service (Threat #52)

Threat: Attackers attempt to use up all the computing resource of the trust evaluator by sending larger number of rule reply.

Mitigation: A firewall can block network access from unauthorised ip address.

Threats against Verify

Tampering (Threat #88)

Threat: Attacker insert malicious codes into the data path between the provenance aware application and provenance record store to hijack the session.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

In addition, most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Information Disclosure (Threat #89)

Threat: Attacker reads the data path between the provenance aware application and provenance record store.

Mitigation: The TPM will have to rely on the security of the TCB to prevent an attacker from inserting malicious codes between the user application and TPM. The integrity measurement of the TCB at system start up can offer protection against this type of attack but there is a risk of run time attack.

In addition, most modern operating systems offer memory protection to prevent a process from accessing a memory location not allocated to it.

Denial of Service (Threat #90)

Threat: Attacker attempts to use up all computing resources by sending large number of replies to the provenance aware system.

Mitigation: The session can be controlled by a log in process.

Data Stores

Threats against Provenance record store

Tampering (Threat #21)

Threat: Malicious entity access the provenance record store and alters the provenance records.

Mitigation: A provenance record is encrypted and the cryptographic key is stored in the TPM. The cryptographic key can be migrated to the TPM of the evaluator and then subsequently used to decrpyt the provenance record for the purpose of trust assessment.

Repudiation (Threat #22)

Threat: Provenance record store denies that it possess a provenance record.

Mitigation: The provenance record store can be an unique ID and this ID is tagged to a provenance record when it is first stored.

Information Disclosure (Threat #23)

Threat: The content of the provenance record store is accessed by an unauthorised party.

Mitigation: A provenance record is encrypted and the cryptographic key is stored in the TPM. The cryptographic key can be migrated to the TPM of the evaluator and then subsequently used to decrpyt the provenance record for the purpose of trust assessment.

Denial of Service (Threat #24)

Threat: Attacker attempts to run provenance record store out of space.

Mitigation: The coding of the provenance record store has to include provision for the scenario of insufficient storage space. The provenance record store can prompt the computer user to remove selected provenance records. The provenance record store should not crash in this scenario.

Threats against Rule store

Tampering (Threat #43)

Threat: Malicious entity access the rule base and alters the trust assessment rules.

Mitigation: The rule base can be encrypted and the cryptographic key can be stored in the TPM. The evaluator will have to provide the correct authorisation value to obtain the cryptographic key from the TPM and then proceed to decrypt the rule base before it uses the rules for a trust assessment.

Repudiation (Threat #44)

Threat: Rule store denies it's ownership of a rule.

Mitigation: Rule store can have an unique ID and this ID is tagged to the rule when it is first stored.

Information Disclosure (Threat #45)

Threat: The content of the rule base is accessed by an unauthorised party.

Mitigation: The rule base can be encrypted and the cryptographic key can be stored in the TPM. The evaluator will have to provide the correct authorisation value to obtain the cryptographic key from the TPM and then proceed to decrypt the rule base before it uses the rules for a trust assessment.

Denial of Service (Threat #46)

Threat: Attacker use up all the computing resource of the rule store by sending large number of access

request.

Mitigation: A firewall can block network access from unauthorised IP address.

Certifications

No certifications have been made for this threat model.

External Dependencies

There are no external dependencies.

Implementation Assumptions

There are no implementation assumptions.

External Security Notes

There are no external security notes.

Bibliography

[1] Masoom Alam, Xinwen Zhang, Mohammad Nauman, Tamleek Ali, and Jean-Pierre Seifert. Model-based behavioral attestation. In 13th ACM Symposium on Access Control Models and Technologies, pages 175–184. ACM, 2008. 50

[2] Drew Amorosi. Solving the TPM uptake challenge. Infosecurity Magazine, September 2013. 2, 14

[3] Alessandro Armando, Gabriele Costa, and Alessio Merlo. Bring your own device, securely. In 28th Annual ACM Symposium on Applied Computing, pages 1852–1858. ACM, 2013. 51

[4] Will Arthur and David Challener. A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security. Apress, 2015. 41, 71, 73, 74

[5] Shane Balfe, Eimear Gallery, Chris J Mitchell, and Kenneth G Paterson. Challenges for trusted computing. IEEE Security & Privacy, (6):60–66, 2008. 49

[6] David Beckett, Tim Berners-Lee, and Eric Prudhommeaux. Turtle-terse RDF triple language. W3C Team Submission, 14, 2008. 56

[7] John Benamati, Mark A Fuller, Mark A Serva, and Jack Baroudi. Clarifying the integration of trust and TAM in e-commerce environments: implications for systems design and management. IEEE Transactions on Engineering Management, 57(3):380–393, 2010. 11

[8] Terry Benzel. The science of cyber security experimentation: The DETER project. In 27th Annual Computer Security Applications Conference, pages 137–148. ACM, 2011. 102

[9] Danilo Bruschi, Lorenzo Cavallaro, Andrea Lanzi, and Mattia Monga. Replay attack in TCG specification and solution. In 21st Annual Computer Security Applications Conference, pages 11–pp. IEEE, 2005. 80, 83

[10] Richard A Caralli, James F Stevens, Lisa R Young, and William R Wilson. Introducing OCTAVE allegro: Improving the information security risk assessment process. Technical report, Defense Technical Information Center Document, 2007. 81

[11] Liqun Chen and Mark Ryan. Offline dictionary attack on TCG TPM weak authorisation data, and solution. In Future of Trust in Computing, pages 193–196. Springer, 2009. 80, 83

[12] Liqun Chen and Mark Ryan. Attack, solution and verification for shared authorisation data in TCG TPM. In Formal Aspects in Security and Trust, pages 201–216. Springer, 2010. 80, 83

[13] David Chisnall. The definitive guide to the Xen hypervisor. Pearson Education, 2008. 103

[14] Fred Chong, Ruby B. Lee, and Claire Vishik. National cyber leap year summit. Technical report, Networking and Information Technology Research and Development Program, 2009. `https://www.nitrd.gov/nitrdgroups/index.php?title=National_Cyber_Leap_Year_Summit_2009`. 1

[15] Karen Clarke. Trust in technology: A socio-technical perspective, volume 36. Springer Science & Business Media, 2006. 9

[16] Cynthia L Corritore, Beverly Kracher, and Susan Wiedenbeck. On-line trust: concepts, evolving themes, a model. International Journal of Human-Computer Studies, 58(6):737–758, 2003. 10

[17] Fred D Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly, pages 319–340, 1989. vi, 10, 11

[18] Grit Denker, Lalana Kagal, Tim Finin, Massimo Paolucci, and Katia Sycara. Security for DAML web services: Annotation and matchmaking. In The Semantic Web-ISWC, pages 335–350. Springer, 2003. 79

[19] Danny Dhillon. Developer-driven threat modeling: Lessons learned in the trenches. IEEE Security & Privacy, (4):41–47, 2011. 80

[20] Tim Dierks and Eric Rescorla. The transport layer security (TLS) protocol version 1.2. 2008. 65

[21] Paul England and Jork Loeser. Para-virtualized TPM sharing. In Trusted Computing-Challenges and Applications, pages 119–132. Springer, 2008. vi, 87, 90, 91, 95, 103

[22] Dieter Fensel, Frank Van Harmelen, Ian Horrocks, Deborah L McGuinness, and Peter F Patel-Schneider. OIL: An ontology infrastructure for the semantic web. IEEE Intelligent Systems, (2):38–45, 2001. 70

[23] Martin Fishbein and Icek Ajzen. Belief, attitude, intention and behavior: An introduction to theory and research. Addison-Wesley, 1975. 11

[24] Susannah Fox and Maeve Duggan. Tracking for health. Pew Research Center's Internet & American Life Project, 2013. `http://pewinternet.org/Reports/2013/Tracking-forHealth.aspx`. 25

[25] Andreas Fuchs, Sigrid Gürgens, and Carsten Rudolph. Formal notions of trust and confidentiality-enabling reasoning about system security. Journal of Information Processing, 19:274–291, 2011. 69

[26] Paul Groth and Luc Moreau. Recording process documentation for provenance. IEEE Transactions on Parallel and Distributed Systems, 20(9):1246–1259, 2009. 103

[27] Thomas R Gruber. A translation approach to portable ontology specifications. Knowledge Acquisition, 5(2):199–220, 1993. 69

[28] Sigrid Gurgens, Carsten Rudolph, Dirk Scheuermann, Marion Atts, and Rainer Plaga. Security evaluation of scenarios based on the TCG's TPM specification. In European Symposium on Research in Computer Security, pages 438–453. Springer, 2007. 80, 83

[29] Vivek Haldar, Deepak Chandra, and Michael Franz. Semantic remote attestation: a virtual machine directed approach to trusted computing. In USENIX Virtual Machine Research and Technology Symposium, 2004. 50

[30] Joseph Y Halpern and Judea Pearl. Causes and explanations: A structural-model approach. Part I: Causes. The British Journal for the Philosophy of Science, 56(4):843–887, 2005. 32

[31] Olaf Hartig. Querying trust in RDF data with TSPARQL. In The Semantic Web: Research and Applications, pages 5–20. Springer, 2009. 102

[32] Zahid Hasan, Alina Krischkowsky, and Manfred Tscheligi. Modelling user-centered-trust (UCT) in software systems: Interplay of trust, affect and acceptance model. In 6th International Conference on Trust and Trustworthy Computing. Springer, 2012. vi, 10, 12, 18

[33] James Hendler and Deborah L McGuinness. The DARPA agent markup language. IEEE Intelligent Systems, 15(6):67–73, 2000. 70

[34] Shawn Hernan, Scott Lambert, Tomasz Ostwald, and Adam Shostack. Threat modeling-uncover security design flaws using the STRIDE approach. MSDN Magazine-Louisville, pages 68–75, 2006. vi, 81

[35] Michael Howard and David Leblanc. Writing secure code, practical strategies and techniques for secure application coding in a networked world. Microsoft Press, December 2002. 81

[36] Michael Howard and Steve Lipner. The security development lifecycle. Microsoft Press, 2006. 23

[37] International Organization for Standardization. ISO/IEC 27001-Information Security Management, 2013. 12

[38] Wayne Jansen. Directions in security metrics research. DIANE Publishing, 2010. 16

[39] Wu Jin, Liao Yongjian, Nie Xuyun, and Liu Mengjuan. The trust management model of trusted software. In International Forum on Information Technology and Applications, volume 3, pages 534–537. IEEE, 2009. 9

[40] Maria Karyda, Theodoros Balopoulos, S Dritsas, L Gymnopoulos, S Kokolakis, C Lambrinoudakis, and S Gritzalis. An ontology for secure e-government applications. In 1st International Conference on Availability, Reliability and Security, pages 5–pp. IEEE, 2006. 68

[41] Anya Kim, Jim Luo, and Myong Kang. Security ontology for annotating resources. In On the Move to Meaningful Internet Systems. Springer, 2005. 30, 78

[42] Dirk Kuhlmann, Arnd Weber, Konrad Eriksson, Thomas Fischer, Stephane Lo Presti, Gianluca Ramunno, Steffen Schulz, Dirk Weber, and Wolfgang Weidner. The evolution of the OpenTC architecture illustrated via its proof-of-concept prototypes. Technical report, Karlsruhe Institute of Technology, Institute for Technology Assessment and Systems Analysis, 2009. 2

[43] Ora Lassila and Ralph R Swick. Resource description framework (RDF) model and syntax specification. World Wide Web Consortium (W3C) Recommendation, 1999. 69

[44] David Lewis. Causation. The Journal of Philosophy, pages 556–567, 1973. 32

[45] Xin Li, Traci J Hess, and Joseph S Valacich. Why do we trust new technology? A study of initial trust formation with organizational information systems. The Journal of Strategic Information Systems, 17(1):39–71, 2008. 13, 22

[46] Gitte Lindgaard, Cathy Dudek, Devjani Sen, Livia Sumegi, and Patrick Noonan. An exploration of relations between visual appeal, trustworthiness and perceived usability of homepages. ACM Transactions on Computer-Human Interaction (TOCHI), 18(1):1, 2011. 13, 23

[47] John Lyle and Andrew Martin. Trusted computing and provenance: Better together. In 2nd Workshop on the Theory and Practice of Provenance, 2010. 51, 66

[48] Ewen Macaskill and Gabriel Dance. NSA files: Decoded. what the revelations mean for you. The Guardian, 2013. `https://www.theguardian.com/us-news/the-nsa-files`. 23

[49] Andrew Martin. The ten page introduction to trusted computing. Computing Laboratory, Oxford University Oxford, 2008. 49

[50] Roger C Mayer, James H Davis, and F David Schoorman. An integrative model of organizational trust. Academy of Management Review, 20(3):709–734, 1995. 13, 22

[51] D Harrison Mcknight, Michelle Carter, Jason Bennett Thatcher, and Paul F Clay. Trust in a specific technology: An investigation of its components and measures. ACM Transactions on Management Information Systems (TMIS), 2(2):12, 2011. 11

[52] Caroline Möckel and Ali E Abdallah. Threat modeling approaches and tools for securing architectural designs of an e-banking application. In 6th International Conference on Information Assurance and Security (IAS), pages 149–154. IEEE, 2010. 85

[53] Luc Moreau, Ben Clifford, Juliana Freire, Joe Futrelle, Yolanda Gil, Paul Groth, Natalia Kwasnikowska, Simon Miles, Paolo Missier, Jim Myers, et al. The open provenance model core specification (v1. 1). Future Generation Computer Systems, 27(6):743–756, 2011. 50, 53, 66

[54] Luc Moreau and Paolo Missier. PROV-DM: The PROV Data Model. World Wide Web W3C Consortium, 2013. vi, 5, 51, 53, 54

[55] Aarthi Nagarajan, Vijay Varadharajan, Michael Hitchens, and Eimear Gallery. Property based attestation and trusted computing: Analysis and challenges. In 3rd International Conference on Network and System Security, pages 278–285. IEEE, 2009. 1

[56] Cornelius Namiluko and Andrew Martin. Provenance-based model for verifying trust-properties. In Trust and Trustworthy Computing, volume 7344 of Lecture Notes in Computer Science. Springer, 2012. 14, 66

[57] Natalya F. Noy and Deborah L McGuinness. Ontology development 101. Knowledge Systems Laboratory, Stanford University, 2001. 68, 70

[58] Leo Obrst, Penny Chase, and Richard Markeloff. Developing an ontology of the cyber security domain. In Semantic Technology for Intelligence, Defense, and Security, pages 49–56, 2012. 103

[59] Open Trusted Computing. VTPM Architecture Revision Final 1.0 Update., May 2009. 87

[60] Bill Parducci, Hal Lockhart, and Erik Rissanen. Extensible access control markup language (XACML) version 3.0. OASIS Std., August, 2011. 60

[61] Bryan Parno. Bootstrapping trust in a "trusted" platform. In Hot Topics in Security, 2008. 85

[62] PCI-SIG. Single Root I/O Virtualization and Sharing Specification, January 2010. 97

[63] Judea Pearl. Causality. Cambridge University Press, 2009. 31, 32

[64] Ronald Perez, Reiner Sailer, Leendert van Doorn, Stefan Berger, Ramon Caceres, and Kenneth Goldman. vTPM: Virtualizing the trusted platform module. In 15th Conference on USENIX Security Symposium, pages 305–320, 2006. 87, 95

[65] Martin Pirker and Johannes Winter. Semi-automated prototyping of a TPM v2 software and hardware simulation platform. In Trust and Trustworthy Computing, pages 106–114. Springer, 2013. 97

[66] Bruce Potter. High time for trusted computing. IEEE Security & Privacy, 7(6):54–56, 2009. 2

[67] Bruce Potter. Microsoft SDL threat modelling tool. Network Security, 2009(1):15–18, 2009. 63, 81

[68] Graeme Proudler, Liqun Chen, and Christoper Dalton. Trusted Computing Platform, TPM 2.0 in Context. Springer, 2014. 71, 73

[69] Eric PrudHommeaux and Andy Seaborne. SPARQL query language for RDF. World Wide Web Consortium W3C recommendation, 15, 2008. 73

[70] NIST FIPS Pub. 197: Advanced encryption standard (AES). Federal Information Processing Standards Publication, 197:441–0311, 2001. 65

[71] Lili Qiu, Yin Zhang, Feng Wang, Mi Kyung, and Han Ratul Mahajan. Trusted computer system evaluation criteria. In National Computer Security Center, 1985. 12, 71

[72] Dan Remenyi, George Onofrei, and Joseph English. An introduction to statistics using Microsoft Excel. Academic Publishing Limited, 2011. 24

[73] Ronald L Rivest. Learning decision lists. Machine learning, 2(3):229–246, 1987. 45, 60

[74] Ahmad-Reza Sadeghi and Christian Stuble. Property-based attestation for computing platforms: caring about properties, not mechanisms. In Workshop on New Security Paradigms, pages 67–77. ACM, 2004. 1, 49, 50

[75] Ahmad-Reza Sadeghi, Christian Stüble, and Marcel Winandy. Property-based TPM virtualization. In 11th International Conference on Information Security. Springer, 2008. 103

[76] Paul Saitta, Brenda Larcom, and Michael Eddington. Trike V. 1 methodology document [draft]. 2005. `http://dymaxion.org/trike/Trike_v1_Methodology_Documentdraft.pdf`. 81

[77] Vincent Scarlata, Carlos Rozas, Monty Wiseman, David Grawrock, and Claire Vishik. TPM virtualization: Building a general framework. In Trusted Computing, pages 43–56. Springer, 2008. 6, 87, 95

[78] Adam Shostack. Experiences threat modeling at Microsoft. In Modeling Security Workshop. Department of Computing, Lancaster University, UK, 2008. 81

[79] Adam Shostack. Threat modeling: Designing for security. John Wiley & Sons, 2014. 103

[80] Tom Simonite. CES 2014: Smart homes open their doors. MIT Technology Review, January 2014. 25

[81] Vaishali Singh and SK Pandey. Revisiting security ontologies. 11, 2014. 71

[82] Bjørnar Solhaug, Dag Elgesem, and Ketil Stølen. Why trust is not proportional to risk. In 2nd International Conference on Availability, Reliability and Security, pages 11–18, 2007. vi, 9, 10, 14, 16, 17, 29, 102

[83] John Steven. Threat modeling-perhaps it's time. IEEE Security & Privacy, 8(3):83–86, 2010. 80

[84] Gary Stoneburner. Underlying technical models for information technology security: recommendation of the National Institute of Standards and Technology. US Department of Commerce, Computer Security Division, Information Technology, National Institute of Standards and Technology, 2001. 31, 69

[85] Frederic Stumpf and Claudia Eckert. Enhancing trusted platform modules with hardware-based virtualization techniques. In 2nd International Conference on Emerging Security Information, Systems and Technologies, pages 1–9. IEEE, 2008. vi, 91, 92, 95, 97, 98, 103

[86] Frank Swiderski and Window Snyder. Threat modeling. Microsoft Press, 2004. 80

[87] C Tarnovsky. Semiconductor security awareness. Today & Yesterday. BlackHat, 2010. 85

[88] Trusted Computing Group. Virtualized Trusted Platform Architecture Specification 1.0 Revision 0.26., September 2011. 87, 94

[89] Trusted Computing Group. TNC Architecture for Interoperability. Specification Version 1.5. Revision 4., May 2012. 38

[90] Trusted Computing Group. TNC IF-MAP Metadata for Network Security. Specification Version 1.1. Revision 9., May 2012. 39

[91] Trusted Computing Group. IF-MAP Metadata for ICS Security. Specification Version 1.0. Revision 46., September 2014. 39

[92] Trusted Computing Group. Trusted Platform Module Library Family 2.0 Level 00 Revision 01.16, October 2014. 2, 12, 49, 63, 68, 71, 82, 88

[93] Frank Van Harmelen and Deborah L McGuinness. OWL web ontology language overview. World Wide Web Consortium (W3C) Recommendation, 2004. 69, 70

[94] Claire Vishik, Anand Rajan, Chris Ramming, David Grawrock, and Jesse Walker. Defining trust evidence: research directions. In 7th Annual Workshop on Cyber Security and Information Intelligence Research, page 66. ACM, 2011. 1

[95] Ye Diana Wang and Henry H Emurian. An overview of online trust: Concepts, elements, and implications. Computers in Human Behavior, 21(1):105–125, 2005. 10

[96] Rolf H Weber and Romana Weber. Internet of Things. Springer, 2010. 19

[97] Stuart Weibel. The Dublin Core: A simple content description model for electronic resources. Bulletin of the American Society for Information Science and Technology, 24(1):9–11, 1997. 53

[98] H Winter. System security assessment using a cyber range. In 7th International Conference on System Safety, incorporating the Cyber Security Conference, pages 1–5. IET, 2012. 102